February 1999

;login:

The USENIX Association Magazine

volume 24 • number 1

inside:

features:

USENIX The Advanced Computing Systems Association

upcoming events

Workshop on Embedded Systems

Co-sponsored by the MIT Media Laboratory

WHEN	WHERE	WHO *program co-chairs*
March 29-31/99	Cambridge, MA	**Dan Geer & Mike Hawley**

DEADLINES

Final Papers
March 15/**99**

1st Conference on Network Administration

Co-sponsored by SAGE

WHEN	WHERE	WHO *program co-chairs*
April 7-9/99	Santa Clara, CA	**David Williamson & Paul Ebersman**

DEADLINES

Final Papers
February 23/**99**

1st USENIX Workshop on Intrusion Detection & Network Monitoring

WHEN	WHERE	WHO *program chair*
April 11-12/99	Santa Clara, CA	**Marcus J. Ranum**

5th Conference on Object-Oriented Technologies and Systems (COOTS)

WHEN	WHERE	WHO *program chair*
May 3-7/99	San Diego, CA	**Murthy V. Devarakonda**

USENIX Workshop on Smartcard Technology

WHEN	WHERE	WHO *program co-chairs*
May 10-11/99	Chicago, IL	**Scott Guthery & Peter Honeyman**

SANS99

Co-sponsored by SAGE

WHEN	WHERE
May 9-15/99	Baltimore, MD

USENIX Annual Technical Conference

WHEN	WHERE	WHO
June 7-11/99	Monterey, CA	**Avi Rubin,** Program Chair **Clem Cole & John Heidemann,** IT Coordinators **Jordan Hubbard,** Freenix Track Chair

3rd USENIX Windows NT Symposium

WHEN	WHERE	WHO *program co-chairs*
July 12-16/99	Seattle, WA	**Werner Vogels & Stephen Walli**

DEADLINES

Paper Submission	Notification to Authors	Final Papers
February 23/**99**	March 23/**99**	June 1/**99**

2nd Large Installation System Administration of Windows NT Conference (LISA-NT)

Co-sponsored by USENIX and SAGE

WHEN	WHERE	WHO *program co-chairs*
July 12-16/99	Seattle, WA	**Gerald Carter & Ralph Loura**

DEADLINES

Paper Submission	Notification to Authors	Final Papers
February 23/**99**	March 23/**99**	June 1/**99**

Eighth USENIX Security Symposium

WHEN	WHERE	WHO
August 23-26, 1999	Washington, D.C.	**Win Treese,** Program Chair **Avi Rubin,** IT Coordinator

DEADLINES

Paper Submissions	Notification to Authors	Final Papers
March 16/**99**	April 21/**99**	July 12/**99**

2nd Conference on Domain-Specific Languages

Sponsored by USENIX in cooperation with ACM SIGPLAN and SIGSOFT

WHEN	WHERE	WHO *program chair*
October 3-6/99	Austin, TX	**Thomas Ball**

DEADLINES

Paper Submissions	Notification to Authors	Final Papers
March 22/**99**	June 2/**99**	August 24/**99**

2nd USENIX Symposium on Internet Technologies and Systems

Co-sponsored by the IEEE Computer Society Task Force on Internetworking

WHEN	WHERE	WHO *program chair*
October 11-14/99	Boulder, CO	**Fred Douglis**

DEADLINES

Extended Abstracts	Notification to Authors	Final Papers
April 15/**99**	May 28/**99**	August 31/**99**

For a complete list of future USENIX events, access *http://www.usenix.org/events*

contents

;login:

;login: is the official magazine of the USENIX Association.

;login: (ISSN 1044-6397) Volume 24, Number 1 (February 1999) is published bimonthly by the USENIX Association, 2560 Ninth Street, Suite 215, Berkeley, CA 94710

$40 of each member's annual dues is for an annual subscription to *;login:*. Subscriptions for nonmembers are $50 per year.

Periodicals postage paid at Berkeley, CA and additional offices.

POSTMASTER: Send address changes to *;login:*, USENIX Association, 2560 Ninth Street, Suite 215, Berkeley, CA 94710.

Editorial Staff

Editor:
Rob Kolstad <kolstad@usenix.org>

SAGE News and Features Editor:
Tina Darmohray <tmd@usenix.org>

Standards Report Editor:
Nick Stoughton <nick@usenix.org>

Managing Editor:
Jane-Ellen Long <jel@usenix.org>

Copy Editor:
Eileen Cohen

Proofreader:
Kay Keppler

Designer:
Vinje Design

Typesetter:
Festina Lente

Advertising

Cynthia Deno <cynthia@usenix.org>

Membership and Publications

USENIX Association
2560 Ninth Street, Suite 215
Berkeley, CA 94710
Phone: 510 528 8649
FAX: 510 548 5738
Email: <office@usenix.org>
WWW: <http://www.usenix.org>

The closing dates for submission to the next two issues of *;login:* are April 6, 1999 and June 1, 1999.

in this issue . . .

by Jane-Ellen Long
Managing Editor
<jel@usenix.org>

Boston in December was a hot place to be. So thought the attendees of the Twelfth Systems Administration Conference, LISA '98. You missed it? Read on: Rik Farrow muses on the conference, Tina Darmohray draws a moral from it, and the Conference Reports offer other perspectives.

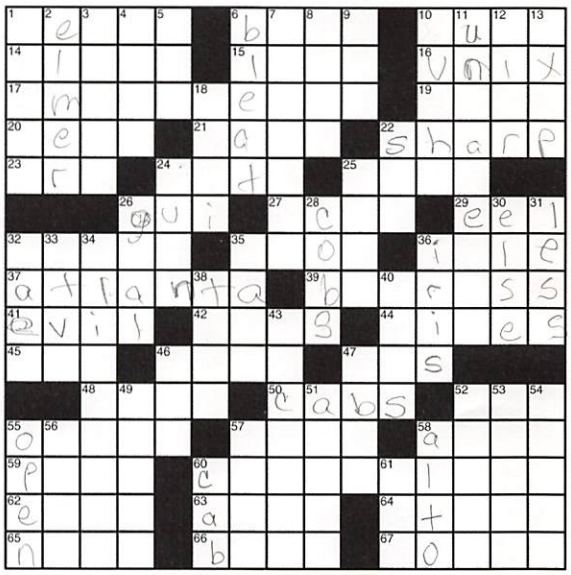
Something new in the SAGE section: Hal Pomeranz introduces our "How-to Notes" series with a clear and concise set of instructions for installing anonymous FTP. Among other offerings, John Sellens goes to the heart of reliability issues: the users themselves – how to communicate with them, how to enlist their assistance, and how to keep them informed. And Hal Miller considers the increasingly complex issues of system security.

In the Features section, Dario Forte continues the discussion of security with a survey of current intrusion-detection systems: how effective are they in monitoring and reporting network attacks? Focusing on Squid as a firewall component, Jeffrey Mogul reports on his experience tracking down and eliminating a major inefficiency in Squid's CPU utilization. On a lighter note, Spike Ilacqua 'fesses up to his role in perpetrating the first ISP.

Before you dive in, try your hand at our latest acquisition, the USENIX Crossword Puzzle.

Across
1. Great software property
6. Splotch
10. Surrounds Dennis's head
14. First test
15. On or in successor
16. USENIX OS
17. Bad sysadmin property
19. God doesn't play these!
20. Appear
21. Encrust
22. Pointed
23. Stray
24. Better test
25. Upon
26. Crutch
27. Telomere locale
29. Moray
32. Burn
35. Deed
36. Lazy
37. Georgia city
39. Retribution
41. Sinister
42. Italian desserts
44. Assistants
45. TV ____
46. Wagers
47. Free TV enablers
48. Utterance
50. Taxis
52. Slow x86 pointer type
55. Incense home
57. The name of the game
58. Too
59. Database operation
60. Why the stock market works
62. Follicle folly
63. Plane downers
64. Dear
65. Manipulates dishonestly
66. Pillow places
67. Network servers

Down
1. Poker action
2. Glue magnate
3. Meth
4. Hoax
5. Corrode
6. Lamb plea
7. Way of being united
8. Former
9. Busy one
10. Low frequency signal
11. Not blocked
12. A staple
13. Released
18. Chilled
22. Greek goddess that fostered craziness
24. Offers
25. Pot
26. Pickling substance
28. Corn units
30. If then ____
31. Replacement for more
32. Loses resiliency
33. Remedy
34. Stirring up
35. Having one of these is not an art
36. Bulb flower
38. Cliche dad presents
40. Progenitors
43. Evaded
46. Fu ender
47. Encourage
49. Ocean routes
51. Faulty
52. Bugs
53. Resource
54. Frolics
55. Doors are sometimes ____
56. Places
57. Front
58. Singing voice
60. Taxi
61. Fraxinus tree



letter to the editor

On Free Software

[*Editor's Note: I received this as private email from Matt, who is a freshman at MIT. He consented to publishing it.*]

Dear Rob:

On reading the latest Microsoft trial news and all, I keep hearing about how Linux is so great. I went over to <www.slashdot.org>, a site that appears to be for and by open-source software (OSS) fanatics, and the things they say absolutely disgust me.

It was always my impression that most Linux fans simply disliked Microsoft. That's okay. But what's not okay, in my mind, is that they seem to go much further than that. They don't just dislike Microsoft. I may be generalizing unfairly, but it seems that these OSS fanatics are opposed not only to specific companies' products but to the entire idea of capitalism.

They keep referring to how OSS represents freedom, how they are paving the way for a world in which capitalism will become obsolete. One went so far as to say that capitalism would be dead in 50 years simply due to OSS ideology. To me, OSS looks like a step backward, not a step forward. These OSS tenets seem just like those of communism all over again, just under a different, more lucrative name and a new breed of supporters. Freedom? Yeah, the freedom to work really hard on something and get no rewards beyond hacker pride and the idea that you've contributed to a "good cause."

I was always under the impression that OSS fans liked Netscape. Well, obviously not. They were attacking the new Netscape/AOL/Sun alliance as if it was Microsoft. It's not Microsoft that they target, it's *all corporations*. This, despite Netscape having actually released its source code for its browser.

One person referred to a future in which people wouldn't work for money but for love and fame and such things. Well, if that's the future, and that's their idea of progress, you can count me out of that future! Working for love and fame is only going to work for about a week before you realize that you can't feed or clothe yourself and sure as hell can't live the "good life"! Certainly no gigantic houses with racquetball courts, swimming pools, or 10-foot widescreen TVs!

I'm tired of OSS fans; maybe they should win, just so they can see how bad their future is. In particular, since a lot of them work day jobs and do OSS as a hobby, I think it would amusing to watch them drive their own companies out of business with their free software and then have to beg for change on the streets: "Starving Linux developer. Will work for food."

Remember the Halloween memo? They all treat it as if Microsoft was being unreasonable in being scared of OSS, that it was proof of their theory that corporations are inherently oppressive of "freedom." Well, maybe it's just me, but I would have thought that there would be a lot of people out there who would completely empathize with Microsoft. People ask how they can compete against Microsoft, but competing against Microsoft is incomparable to competing with the Linus Torvalds of the world, who (1) don't want to make a profit from their work and (2) don't have to pay their "employees." If I were Linus Torvalds, I'd be hitting myself *really* hard for somehow managing to make a product used by millions of people and yet somehow having managed to not get rich off of it.

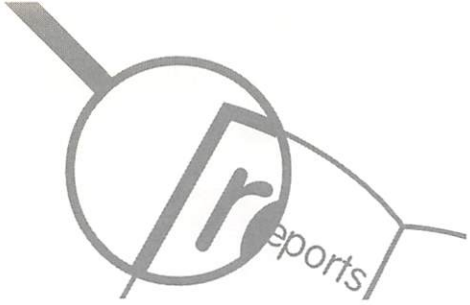
Did I mention that it appears they want to make intellectual property illegal (or at least some of them do)? I bet they wouldn't be so enthusiastic about a change like that if they actually thought it through and realized that it would mean that there would be zero incentive to create intellectual property, since everyone would just rip it off. First effect, no more media companies. TV would die. Radio would die. The commercial parts of the Internet would die. Many books would never be published. Then no more software companies. What company would want to create a software program they had no rights to? Red Hat and others have a working business model simply because they don't *have* to create Linux. Others do it for them, and they sell support. But if Red Hat had to develop Linux itself, they'd be bankrupt in six months.

In other words, their prescription for freedom of information would certainly free information to be used by all, but it would also backfire and make it so that no one would ever want to generate new information!

Making this all the more real and frightening to me is the fact that MIT is the place of origin of OSS, and that its students seem to be the biggest proponents of it around. There are a horrifyingly large number of Linux fanatics among the students here.

If this is the way our economy is going in the future, count me out of it. Give me plain old capitalism any day.

Matthew Craighead
<craighea@MIT.EDU>



conference reports

This issue's reports focus on the Twelfth Systems Administration Conference (LISA '98), held in Boston, MA, on December 6-11, 1998.

Our thanks to the summarizers:

Chastity D. Arthur

Kurt Dillard

Carolyn M. Hennings

Brian Kirouac

Douglas Stewart

Allen Supynuk

John Talbot

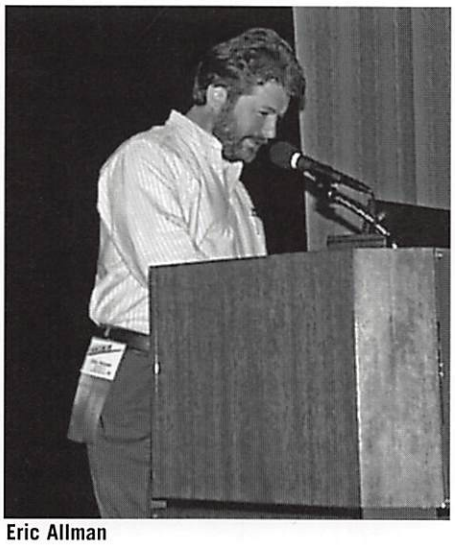
Bruce Alan Wynn

KEYNOTE ADDRESS

The Evolution of Open Source Software

Eric Allman, Sendmail, Inc.

Summary by Carolyn M. Hennings



Eric Allman

The original author of sendmail, Eric Allman, has been actively involved in all aspects of software development. He provided an authoritative view of the history, current state, and possible future of what we now call open-source software.

Allman reminded us that the first implementations of computer systems were mainframes that required special facilities and staff. These systems came with the source code for the operating system, and all configuration information was in the source code. The next systems to become available were minicomputers designed originally for lab environments. These systems were less expensive and required less support, but the users were more sophisticated. Some commercial software started coming on the scene, but most software was free, and the source code was available. With both the mainframe and minicomputer systems, the hardware blueprints were available.

The advent of microcomputers changed the characteristics of the computer-user community. Microcomputers were affordable and available to the hobbyist. As the hardware became more and more available and software became easier and easier to use, the demand for systems increased. With the availability of cheap, reliable hardware, the need for blueprints and specifications decreased. At the same time, the demand and need for knowledge about the internal workings of the computer software decreased. These factors led us to our current situation of cheap hardware and expensive software.

The recent movement toward open-source software has been built on the desire and contributions of avid technologists who were frustrated by the inability to access, understand, and optimize the software. The expansion of the Internet and the resulting ability quickly and easily to share code has provided a forum for this community to develop software on "Internet time."

Allman referred to "The Cathedral and The Bazaar" article <http://www.tuxedo.org/~esr/writings/cathedral-bazaar/> by Eric S. Raymond comparing the commercial centralized software-development process to a cathedral and the decentralized development of open-source software on the Internet to a bazaar. Allman suggested that a moderator is necessary for truly successful open-source software development. The "bazaar" analogy does not extend to this level. He described a number of new company models supporting the open source philosophy in different ways:

- Give away old source code, sell new.
- Develop and give away source code, sell service.
- Market and give away source code, sell service.
- Develop and give away base set of source code, sell ease-of-use extensions, other tools, and service.
- Develop and give away parts, sell most as closed source.

- Develop and give away source, place restrictions on the subsequent use of the code.

Allman's answer to the question, "Why give it away?" is that "religious" arguments don't make good business. Sendmail, Inc. continues giving away the source code in order to maintain market share, ensure universal interoperability, and improve the quality of the code. He commented that the volunteer individuals likely to support and contribute to open-source code development are naturally focused on the "interesting" and "cool" work. However, commercially viable software requires development that might be less exciting, such as GUIs for configuration and/or management tools. His proposal is to develop and sell these extensions as a commercial package, while continuing to release the base software in the open-source arena.

Allman predicted that features-based code will tend toward open source, while algorithm-based code will generally stay closed. He also suggested that open-protocol-based code will tend toward open source and that, subsequently, mission-critical applications will tend toward open protocols.

REFEREED PAPERS TRACK

Session: Security

Summary by Kurt Dillard

TITAN

Dan Farmer, Earthlink Network; Brad Powell, Sun Microsystems, Inc.; Matthew Archibald, KLA-Tencor

Titan is a freely available host-based tool that can be used to audit or improve the security of a UNIX system. It started as a Bourne Shell script to reconfigure various daemons. Checks for verifying configurations were added, and over time Titan became an effective tool for auditing computers. The authors made it clear that this is a powerful tool not designed for the weak or timid sysadmin. Using it incorrectly, you could easily render a system unusable or even unbootable. For the SA willing to put in the time to learn Titan thoroughly, it can save a great deal of time while helping to verify and maintain security across multiple hosts. The authors also made it clear that Titan is not the be-all and end-all of information-systems security; it is designed to be only part of the overall infrastructure. Titan now runs on most versions of Solaris, but it shouldn't be too difficult to port the scripts to other flavors of UNIX. By editing the scripts you can reconfigure Titan so that it performs auditing and configuration changes appropriate to the type of host you are running it on and the security policies that your network requires.

See <http://www.fish.com/security/titan.html>.

Infrastructure: A Prerequisite for Effective Security

Bill Fithen, Steve Kalinowski, Jeff Carpenter, and Jed Pickel, CERT Coordination Center

The authors started their presentation with some scary data compiled by CERT. A 1997 survey shows that 50% of systems were not kept up to date with security patches *after* they were compromised. One site appeared in 35 incidents between 1997 and 1998; the site was used for password sniffing and probing of other sites in many of those cases. Ten of the 35 incidents involved root compromise of the host. In another break-in, 20-25 hosts were compromised. All of these systems needed to be rebuilt, but the site's administrator said that they didn't have enough resources to do so. The authors set out to improve infrastructure manageability at CERT by creating an easily maintained system of distributing software packages. The result is SAFARI, a centralized repository of 900 collections of software for multiple versions of UNIX. Using SAFARI, a sysadmin can build new systems from scratch and update existing systems with patches and new packages. SAFARI includes flexible version controls so that developers and admins can easily post and retrieve software packages from the same central repository.

SSU: Extending SSH for Secure Root Administration

Christopher Thorpe, Yahoo!, Inc.

Christopher Thorpe needed to create a low-cost method of allowing distributed access to privileged operations. As a system administrator at Harvard University, he had 200 systems to manage, and many of the students, staff, and faculty required root access to one or more processes on those systems. A method was needed to allow these users to execute certain processes as root in a secure environment. SSU combines SSH and Perl to create a system that allows this by combining RSA key pairs with those commands. Unfortunately, users need a separate key for each command that they need access to, but the solution works at Harvard because most of the users need to execute only a few commands with root access. A side-benefit of SSU is that everyone is now using SSH for all of their console connections, making all network activity more secure.

Session: Pushing Users and Scripts Around

Summary by Allen Supynuk

System Management with NetScript

Apratim Purakayastha and Ajay Mohindra, IBM T.J. Watson Research Center

NetScript is a BASIC-like (could also be Perl- or Tcl-like) scripting language for remote administration of heterogeneous systems (UNIX, Wintel, and soon PDAs like the PalmPilot). It includes neat features like parallel scripts, isolation, and disconnected operation.

Single Sign-On and the System Administrator

Michael Fleming Grubb and Rob Carter, Duke University

This presentation did a nice job of covering the major interpretations and issues involved in single sign-on, but, not surprisingly, was a bit short on satisfying solutions. The paper is worth reading for anyone interested in this holiest of grails.

Session: Storage Performance

Summary by Carolyn M. Hennings

Using Gigabit Ethernet to Backup Six Terabytes

W. Curtis Preston, Collective Technologies

Curtis Preston presented his paper and some interesting additional information. He talked about two different backup systems he implemented and some of the things he learned along the way. One conclusion from these experiences was that the limitation in backing up this amount of data in a reasonable timeframe was really in the network. He suggested that private "storage-area networks" will be the future direction.

Configuring Database Systems

Christopher R. Page, Millennium Pharmaceuticals

Christopher Page described relational databases from a system-management viewpoint. Relational databases reside on computer systems that are managed by system administrators. The system administrator needs to be knowledgeable about how the database works with, and uses, the operating system. Page began with information about the relational-database architecture. From a database user's perspective, data is conceptualized in tables; it is accessed and controlled through the Structured Query Language (SQL), and data manipulation is transaction-oriented. From the relational-database server perspective, it is an operating system on top of the native operating system, it manages the steps in processing requests, and it maintains four different file types (data, log, temporary, and control files). Issues that system administrators need to be aware of and work with the database administrators on include: setting up and monitoring memory structures; optimizing "Intimate Shared Memory" (ISM); and configuring swap space, CPU, and network usage. Page noted that some key things to be aware of are data-block size, raw versus filesystem disk usage, and synchronous I/O.

Session: Distributed Computing

Summary by Carolyn M. Hennings

Design and Implementation of an Administration System for Distributed Web Server

C. S. Yang and M. Y. Luo, National Sun Yat-Sen University

This presentation, winner of the Best Student Paper award, described a system for managing distributed Web servers. The components of the system include a control interface, a controller, broker, agent, and remote console GUI. The control interface is used by system administrators for turning distribution on and off, adding and removing nodes, managing Web content, and reading statistical information. The controller is a Java application that runs in the background on the distributor and responds to system-administration requests. The broker is a standalone Java app on each server that consists of an agent and monitoring thread. The agent performs the delegated task on requested nodes such as adding and deleting files, searching for a file, and analyzing log files. The remote console is used for executing management operations and interacting with the controller. The presenters described the system as extensible and applicable to any Web site.

Session: Networking

Summary by John Talbot

MRTG – The Multi Router Traffic Grapher

Tobias Oetiker, Swiss Federal Institute of Technology

MRTG is one of the more valuable free tools in use today in thousands (a very conservative guesstimate) of network and Internet sites around the globe. Tobias Oetiker, MRTG's creator, maintainer, and chief cook and bottlewasher, originally created MRTG in the summer of 1994 using his free time on a hobby project to analyze network traffic at the Montfort University. In the short time since then, MRTG has established its place as an essential tool for network monitoring.

Unfortunately Oetiker is ending his support of MRTG because of personal and professional time constraints. He delivered this news with a touch of sarcasm and modesty by stating that he "sometimes wonder[s] if no one can program" since it is "just a Perl script." Oetiker has done an excellent job of developing and maintaining the MRTG code base, and his decision will mean a sad loss for the progression of this marvelous tool.

For the meat and potatoes of the technical discussion, Oetiker concentrated on new improvements to the round-robin database (RRD) management of the latest release of MRTG (called MRTG-3). RDD enhancements include the ability to store multiple data sources in parallel and a break between the database-storage and the graph-generation interface, which is now called `rrdtool`. He noted that these improvements alone have greatly improved the performance of the MRTG data-logging capabilities.

There was some discussion about using other databases, such as Oracle, to manage the databasing (RRD) and graphical functions (`rrdtool`) of MRTG for large datasets and networks. Oetiker was not

sure if this would provide any major performance advances over the current method. RRD uses a "lossy database" topology in which older data is statistically averaged over spreading sample rates as data ages. Using this method of data management, the RRD is able to maintain a fixed-sized dataset.

While the latest performance enhancements are great news about advancement in the MRTG package, it is sad news that Tobias is limiting his future involvement with MRTG.

Wide Area Network Ecology

Jon T. Meek, Edwin S. Eichert, and Kim Takayama, American Home Products Corporation

This was a good nuts-and-bolts discussion of how to improve your WAN performance. Jon Meek and his team concisely and effectively described practical and innovative solutions for analyzing and enhancing network performance at the American Home Products Corp.

Monitoring basics, such as tcpdump and Perl scripting, were some of the methods used to measure WAN performance of the round-trip times (RTT), committed information rates (CIR), and reliability of these parameters across numerous WAN direct- and virtual-circuit connections. For more unusual problems, Meek and his team looked directly at packets and protocols on the WAN and performed system monitoring of process utilization and file sizes to gather more process data. Often, network-performance measurements and diagnostics were limited in their detail by the complexity of the private section of the frame-relay cloud of their WAN provider. Some solutions to this dilemma were to obtain circuit and network maps from their frame-relay provider(s) and insist on read-only access to the SNMP port of the frame-relay routers to be able to quickly map and identify stops and bottlenecks in the WAN architecture.

Other core essentials to network performance were also checked. Redundant successive database queries, large PostScript files, email attachments, file sharing over the WAN, and SQL network applications were major sources of bandwidth utilization. Hard disks were implemented on many of the network printers to handle redundant headers and footers, such as company logos and graphics. This was a particularly novel idea since it enabled local caching of redundant data at the destination point (the printer). It was found that Web applications used far less bandwidth than the SQL applications, since the Web applications needed to transmit only the interface instead of a client/server database link.

Meek and his team took advantage of a packet prioritization determined by protocol and "weighted fair queueing" to lessen the severity of "bandwidth hogs." Although the use of access-control lists (ACLs) and special packet handling introduces extra load on the network routers, it enabled interactive network sessions to take precedence in the network bandwidth, thereby giving the appearance of better response by online applications such as Telnet and Web interfaces.

In one instance, a WAN circuit was upgraded from a CIR of 128KBps to 256KBps and the RTTs markedly increased, degrading network performance. It was found that the frame-relay provider actually routed the new connection upgrade over a more complex set of routes in the WAN just to get the "faster" circuit connected to Meek's site. If Meek's team did not statistically and periodically analyze network-performance links, they would have had no idea where the trouble was, since their frame-relay provider did not do any network performance monitoring and analysis, only up/down-time status. Meek's analysis data was enough to give his frame-relay provider the impetus to get working on the solutions. Meek stated that many frame-relay providers have this same problem.

Further analysis of WAN RTTs and RTTs of Internet-bound connections showed a much lower RTT for Internet connections than for WAN connections. This left a few questions about the possibility of using Virtual Private Network (VPN) technology routed over the Internet to handle some of the currently poor-performing WAN links.

Automatically Selecting a Close Mirror Based on Network Topology

Giray Pultar

Giray Pultar discussed issues involved in automatically redirecting queries to an HTTP server on the basis of an HTTP client's proximity to the nearest HTTP mirror server.

Automated mirror selection would be of great value to sites that have multiple Internet presence points across large geographical areas. An automated mirror-selection service or tool would provide a single-presence appearance to the client host and greatly reduce the need for a user to manually choose a mirror site from a lead page or site.

Giray made suggestions for implementing such a system and noted some defects of both client- and server-side implementations of close-mirror selection. Java (software overhead) and traceroute (routing difficulties due to propagation delays and return-trip connections to the client host through an origin firewall) are limited as client-side solutions. In addition, the traceroute method could conceivably add large delays for sites with numerous mirroring sites.

Giray's approach to solving the close-mirror problems is to build a mirror table of known networks and relate them to the geographically "closest" mirror. Such a table, if based on all IP network address combinations, would be massive and difficult to construct. How can each network be correctly identified and categorized? How can physical "closeness" be determined when so many ISPs have

multiple redundant links and dynamic failover routes?

By identifying collections of networks as autonomous systems, the definition of "which mirror to use" becomes less complicated. The Internet routing registry (IRR) databases are then used to compile a correlation table which a server can use to redirect a client to a "closer" site based on the client's IP number. Two scripts were developed for building the close-mirror tables. Script `closest.cgi` is the CGI interface to be called by the server to determine the "nearest" location. The script `mkmirrortable` contacts the IRR databases and compiles the correlation table.

Some caveats still apply to the development of this technology. There were questions about "what should be" and "what is" when using AS paths, since there is no hard relation between an AS path and a real physical network route. Also, expansion delays depend on the geographical span and size of a particular AS, which can vary in real physical size and distance within the definition of the AS itself. Other issues, such as load balancing and mirror-site downtime, pose several challenges to the development of this new technology which, once solved, will prove to have highly useful applications not primarily limited to HTTP redirection.

The close-mirror package can be found at <ftp://ftp.coubros.com/pub/lisa98>.

Session: Infrastructure

Summary by Carolyn M. Hennings

What to Do When the Lease Expires: A Moving Experience

Lloyd Cha, Chris Motta, Syed Babar, and Mukul Agarwal, Advanced Micro Devices, Inc.; Jack Ma and Waseem Shaikh, Taos Mountain, Inc.; Istvan Marko, Volt Services Group

Chris Motta related his experience in moving approximately 1,000 machines and 220 users from a single building into two different buildings. He listed some of the things that helped the move go smoothly. This list included a well-defined scope, using email to communicate during the planning phase, organization and planning, a central command center, allowing extra time for unforeseen problems, blanket purchase orders with key vendors, new networks staged and tested in advance, and insisting that managers and users were not present during the move. Some things that hindered the operations were insufficient checking of scripts and lists, inaccurate audit from a vendor, poor estimate and execution from movers, poor estimate/execution from a fileserver vendor, lack of working phone lines, weekly bureaucratic meetings too far in advance of the move, and not getting enough sleep during the move itself. Motta made the following suggestions: Have independent verification of scripts and audit; have a single person in charge of controlling the entire move and making key decisions; have laptops with network cards available for use as terminal emulators; have cellular phones and/or radios available for everyone, and plenty of spare cables and adapters.

Anatomy of an Athena Workstation

Thomas Bushnell, BSG; Karl Ramm, MIT Information Systems

Thomas Bushnell described an academic-computing environment with approximately 30,000 users and 1,000 workstations. The workstations are located in public clusters – libraries and hallways – as well as in faculty and staff offices and dorm rooms. They are all standard UNIX workstations configured for a single user and serial reuse. Bushnell described the concept of "lockers" – storage areas specified for a particular use such as a home directory, packages of software, or common areas for collaborative efforts by groups of people. The "lockers" support the release cycle for operating systems and software updates. A group of "system packs" made up of "lockers" comprise the operating system and other software layered on top of the OS. Machines are identified as parts of clusters that determine at what point in the release cycle new software will be loaded. An "auto-update" facility allows for these lockers of software to be loaded as the machines are booted. The presenters concluded with the following observations: the security model gives clarity; the serial reuse model presents problems with time sharing and long-running batch jobs; and the hands-off auto-update and installation allows a team of 10 system administrators to support the 30,000 users and 1,000 workstations.

Bootstrapping an Infrastructure

Steve Traugott, Sterling Software and NASA Ames Research Center; Joel Huddleston, Level 3 Communications

Steve Traugott provided an insightful overview of the steps for creating and managing a solid infrastructure. Some key steps are determining how version control is going to be done and setting up a "gold server." With this foundation, other infrastructure elements such as installation tools, directory and authentication services, network fileservers, client file access, and configuration management can be implemented. These steps and others are detailed in the paper along with a graphic describing the order in which they should be performed. This architecture has advantages in disaster recovery, software distribution, and lowering total cost of ownership. Traugott concluded the presentation with the observation that when it comes to defining an infrastructure, the role being filled is larger than "system administrator" and might more accurately be called "system architect."

Session: Distributing Software Packages

Summary by Chastity D. Arthur

mkpkg: A Software Packaging Tool

Carl Staelin, Hewlett-Packard Laboratories

Carl Staelin has developed a remarkable tool to allow software publishers to easily create installation packages. Staelin pointed out that the industry has focused on the end users and systems administrators, allowing them to easily install and uninstall software, and has not focused on the first step, the software distributor who has to create the binary installation package. With Staelin's tool, `mkpkg`, the software distributor can add a description of the package, develop manifests, include certain dependencies, create install and uninstall scripts, and customize the post-installation. `mkpkg` can take as little as three minutes to complete, provided the software is ready for distribution.

Staelin addressed the portability of `mkpkg`. It was developed on HP-UX and uses HP-UX-specific commands. He has successfully ported mkpkg to ninstall, update, and SD-UX. His next conquest will be RPM, but his work has slowed for lack of time. `mkpkg` is available at <http://www.hpl.hp.com/personal/Carl_Staelin/mkpkg>.

SEPP – Software Installation and Sharing System

Tobias Oetiker, Swiss Federal Institute of Technology

Tobias Oetiker and his IT support group (ISG) saw the need to provide a software-installation tool that would reduce the repetitive task of installing software and configuring the systems throughout the various departments in the institute. They were also looking to develop a tool that would retain some independence in the installations. The ISG tested software-distribution tools already on the market, comparing Red Hat's Package Manager, GNU Stow, Depot-Lite, and LUDE, to name a few. None of these tools met their requirements, nor did they use wrapper scripts – but the ISG did discover that in a mix of all these tools and a few of their own ideas lay exactly the features they needed. Thus SEPP came into existence. SEPP provides both a clean system for system managers to use and a user-friendly environment. It is currently supported only on Solaris and Irix.

SEPP includes a number of system-management features. The subdirectory tree provides clean encapsulation to all files of the same distribution; a special directory (called SEPP) in each software subdirectory houses a description of the contents along with the startup wrapper script, `start.pl`; the automounter tool, using `/usr/pack`, helps to ensure paths during compiles; the packages' binaries are actually symbolic links to `/usr/sepp/bin`, which points to stub scripts; Perl scripts start up the wrapper script; and a unique name field is generated for each software-package distribution. Oetiker was not only very proud of SEPP's system-management features, he also highlighted the user features. One of SEPP's most convenient features is that the user only needs to add `/usr/sepp/bin` to the PATH variable. The ISG also developed both Web-based and manual-page-based SEPP documentation. SEPP also allows for multiple versions of the same software distribution using suffixes appended to the executable names.

With SEPP's reliance on the automounter, user applications that are required during bootup will cause problems. The ISG is currently addressing that issue by adding a feature that enforces the bootup applications to be mirrored to the local machine.

SEPP is distributed under the GNU General Public License and can be obtained from <http://www.ee.ethz.ch/sepp>, where there is also information on the SEPP mailing list.

Synctree for Single Point Installation, Upgrades, and OS Patches

John Lockard, University of Michigan; Jason Larke, ANS Communications

Synctree is a system-administration tool developed for a large network requiring frequent OS or software updates and security patches. Lockard and Larke had two goals in mind – system security and uptime. Synctree is capable of holding the network's complete configuration in a secure, readable format. The idea behind Synctree is to bring a machine up on the network and "sync" it to the templates for the architecture so defined.

Comparing Synctree to cloning, the authors stated that although a clone could be made that meets your requirements, each time an update is added to that architecture a new clone would have to be established. Another comparison was made to rdist in that rdist relies on each machine being up and connected to the network when you run your update. Under the direction of Paul Howell, the University of Michigan's Computer Aided Engineering Network group wanted to create a utility that provides verification of widely distributed patch installations and ensures that files prone to hackers are in their expected state.

Synctree's template permissions are based at the client level, and only the root user of that client can call a sync and order the classes the client syncs to. Synctree relies on a server, and any other work is copied in downloads to the clients. Synctree also allows images to overlay each other, like GNU's cfengine. With this feature, the client actually builds the final picture before implementing any changes. Synctree has only one configuration file, `/etc/hostconfig`, where each class of machines is listed. Synctree goes down to the level of file-to-file comparison.

In closing their presentation, Lockard and Larke talked of future features they would like to incorporate into Synctree. One update for the near future is allowing Synctree to install software packages normally found on the network to the local hard drive. Currently Synctree relies on AFS, which not everyone has or wants; a future goal is to adapt Synctree to another secure copying system, such as krcp.

A Synctree sample is available for non-commercial use only at <ftp.math.lsa.umich.edu/pub/Synctree/>.

Session: Mailing Lists

Summary by Brian Kirouac

Mailman: The GNU Mailing List Manager

John Viega, Reliable Software Technologies; Barry Warsaw and Ken Manheimer, Corporation for National Research Initiatives

Have you ever subscribed to a list and later realized that you forgot what type of mailing list it was and how to unsubscribe? As a list owner it would be nice to add a footer to each message that describes the process for unsubscribing. Viega, Warsaw, and Manheimer wanted to add just such a footer to a majordomo mailing list. This worked fine for individual messages. The problem was each message with its footer was put in the digest, so there were multiple copies of the footer in the digest.

They started looking at different mailing-list-management software packages for something that would allow the user to subscribe or unsubscribe quickly and easily, and would allow the list owner to manage the list. MajorCool was considered "cool" but limited.

Mailman came of this. Mailman offers a Web-based user interface that allows list management on three levels: user, list, and site. It includes email-based commands, but the Web based interface is the driving force. A user or owner can subscribe or unsubscribe from a list as well as choose between live and digest modes. A list owner can edit the list's Web page and set various list options.

Drinking from the Fire(walls) Hose: Another Approach to Very Large Mailing Lists

Strata Rose Chalup, Christine Hogan, Greg Kulosa, Bryan McDonald, Bryan Stansell, Global Networking and Computing, Inc.

Strata Rose Chalup presented the authors' experience moving the "Firewalls" mailing list. The original server used large ISPs to do mail relaying instead of doing the delivery itself. When GNAC took over the list, it did not have the same relationships with ISPs, and thus the new server had to deliver the mail.

The typical two-queue system did not function well enough. The outbound queue was growing faster than mail was getting delivered. The problem was that majordomo was creating a single sendmail queue file generated with 4000+ addresses in the RCPT line.

They created a Perl program run every five minutes out of cron, called `qsplit`. This takes the original queue file and splits it up into easier-to-spool chunks. Each chunk having a specified number of recipients, they chose 25. To keep the uniqueness of queue file names, each chunk has a sequence number appended to the original name. These are then spread through 10 different queue directories.

Each queue directory runs a separate instance of sendmail to process the queue. A process called spawn is responsible for keeping these sendmail processes running. Spawn is smart enough to keep the system busy but not have it swapping. This way as much mail is delivered as fast as possible.

Request v3: A Modular, Extensible Task Tracking Tool

Joe Rhett, Navigist

This was a presentation on some of the modifications and extensions that have been made to Request.

Some of the problems of not having a good tracking tool are: task history is usually stored in human RAM, thus prone to loss; handoffs are not always handled well; there's little or no information to justify staff. Requirements of a good task tracking tool are: track entire history of task; do not slow down admins who are using the tool; support almost any operating system or platform; work well from remote; be easy for untrained users to access; acquire statistics.

Commercial applications may fit some of these requirements, but they are expensive, require a lot of training, and don't usually support all platforms. Free applications are generally not updated often, require UNIX-like skills, and don't always have Web and email interfaces.

The previous versions of Request had several problems. First, they were not year-2000 compliant. Parts of the code aggregated Perl 5, and small changes required many fixes. Most problems related to dispersion. The design goals of the new request were to fix these problems and to allow others to add code easily. It actually resulted in fewer lines of code.

INVITED TALKS TRACK

Zero to LISA in One Year

Brent Chapman, Covad Communications Company

Summary by Chastity D. Arthur

Brent Chapman explained the successful and unsuccessful decisions made as the Silicon Valley startup, Covad, coped with its one-year growth from one region with 50 people to six regions with 400 people. He was a member of the IT department, faced with continually scaling and supporting the network and responding to systems demands. He discussed the ongoing process of planning every detail possible and attempting a proactive approach to situations. "No plan survives reality, but it's a start," he said.

A startup company must recognize the challenges, both obvious and hidden. The obvious issues are: keeping up with the growth; getting ahead of the growth; attracting and retaining top talent; developing adequate and scalable systems; maintaining daily development support; introducing new tools, services, and concepts to the users; and developing a strong infrastructure not just for IT but for the entire company. Some hidden challenges are: the linear rise in number of hours required; growth of users' expectations; loss of volunteers in the IT department; and old users being more self-sufficient than newer users. In Covad's case, company culture was also a factor. When Covad was smaller, communication was always at peak performance; as the company grew, disseminating information became more of a challenge, and it became harder for older employees to find time to work with new ones.

Chapman discussed the considerations that went into site selection (Silicon Valley offered many advantages) and described the headaches and heartaches of surveying site after site. He explained how IT planned for the move once a location was chosen. What better way than to set up a MOCR, a NASA-like Missions Operation Control Room. Chapman described in depth how the success of the MOCR enhanced the success of Covad's move. The first decision was to hire trusted contractors to augment the staff. Chapman then armed everyone with a radio and appointed flight directors in a rotation that meant the MOCR always had a manager. The flight plan was simple – make all critical decisions in advance. A successful move was completed in one weekend, and there was still time to leave welcome packages in the cubicles. IT left a welcome note for the employee, a map of the new building, including restrooms and printer locations, plus a little treat. The MOCR remained open as a help desk through the first official busy day. Chapman described the MOCR as a "great centralized success."

Chapman then discussed the one demand that kept arising – users wanting more bandwidth. Although bandwidth is the first to be blamed, it is often not the problem. IT should help the executives understand the cost of more bandwidth, help the users understand how to use software or choose their software, and explain the differences between latency and bandwidth.

Chapman ended his talk with why he chose to join Covad as a startup company. He wanted to be a part of something that could be successful and to have the opportunity to work with outstanding and experienced people, interesting technology, and vendors. He closed with the observation that maximum productivity is nowhere equal to the maximum number of hours worked.

Got LDAP? Deploying and Using the Lightweight Directory Access Protocol

Leif Hedstrom, Netscape Communications Corporation

Summary by Brian Kirouac

Is your directory information starting to become overwhelming? Someone suggests LDAP, the Lightweight Directory Access Protocol, so you start looking at the documentation. One of the first things to hit you is that deploying and managing a directory server is a complicated task.

Leif Hedstrom of Netscape gave a good talk dealing with some of the issues and pitfalls associated with installing a new LDAP-based system, based on some of the issues Netscape faced when installing its LDAP servers.

Before designing anything, you need to establish your goals. The first is easy dissemination of information. Two other concerns should be the scalability and performance of your server. Scalability and performance have a direct impact on how to design your database tree structure.

During the design and implementation, you need a manager who will back you, and you need to make sure you have all departments involved. It was amazing how much input the legal department had in Netscape's implementation. Several legal concerns can influence what data you might be able to include. Pictures, home phone numbers, and car license plates, for example, might be problematic.

Useful guidelines for planning and implementing LDAP: spend time planning, analyzing, and testing the design; select a directory-information tree that is as simple as possible; elect the proper software based on your needs.

Succumbing to the Dark Side of the Force: The Internet as Seen from an Adult Website

Dan Klein, Cybertainment, Inc.

Summary by Bruce Alan Wynn

Dan Klein gave effectively the same talk as he had at USENIX '98 in New Orleans, without displaying any defensiveness about the fact that he is the technical person for a dozen pornography Web sites. He went over some of the technical issues for maintaining such a site, and noted that porn sites tend to have better security and adult-verification than some banks. The talk was very well attended. (No, he didn't show pictures; the talk was PG-13.)

On the technical side, Klein talked about techniques to reduce the load on a Web server: load sharing, load shedding, and load boosting. Load sharing is basically using DNS entries in a round-robin fashion to distribute the load. The main issue with this is making sure that all of the servers have the same data. Load shedding requires a front-end server that hands off initial requests to back-end servers that have the real content. The problem here, again, is keeping everything in sync. Load boosting is performed on the client side. A lot of sites make their money based on the number of hits a given URL receives. Thus sites will have banners and Javascript programs that pop up other windows that access the same URL. Load boosting consists of turning off Javascript on the client to prevent these other windows from appearing, thus reducing the amount of time it takes to load a page.

A good practice he mentioned is keeping logs. Logs help plan for the future, and they help determine possible security breaches. And, in the case of legal action, they can help cover you if someone falsely accuses you of something.

Branchstart – A Generic, Multi-OS Installation Server

Rory Toma, WebTV Networks, Inc.

Summary by Chastity D. Arthur

Rory Toma described his successful implementation of a single-architecture yet multi-OS network installation server on Intel-based platforms. His project isn't actually named Branchstart; he is playing on the name of Sun's product, Jumpstart. Toma calls his implementation MOSIP, an image- and package-based OS installer, successfully tested on Red Hat Linux 5.2, NT 4 Workstation, NT 4 Terminal Server, and Windows 95.

Toma's project goals included: minimal user interaction, 100 percent predictability, easy scalability, and functionality at a junior level. He commented that MOSIP is reproducible, flexible, and fast to install. On the more technical side, MOSIP has a binary failure mode; operators can use the same install server for multiple OSes or platforms; and a serial console or GUI is optional. Not so inviting is the amount of front-end work and the level of knowledge and experience needed to set up advanced installations.

Toma described how he made MOSIP come together. For each OS base needed, a template machine must be installed to acquire the OS image with dd and to record software- or hardware-specific parameters. He then described what he terms "laying down the bits" – basically, booting Linux with NFS root filesystem and having an installation script run automatically. Toma chose to replace init with his own script. To finish, he described "modifying the bits." This is the point at which the administrator would modify the IP address, create auto-login scripts, and install LILO.

In closing, Toma discussed his next project, Internet OS Installation Server Project (IOSISP), which will take MOSIP to the next level: installation of free OSes from the Internet. He plans to automate the installation of MacOS and Windows 2000 to include Active Directory and Exchange. He would like his project to function on nonIntel hardware, specifically for NT and Linux. His greatest challenge, he said, is creating a generic NT image that would allow modification to suit a wider range of hardware.

To learn more about MOSIP or IOSISP, visit <http://www.munitions.com/rory/MOSIP/welcome.html>.

THE GREAT CERTIFICATION DEBATE

Moderator: Rob Kolstad. Panelists: Phil Scarr, GNAC; Scottie Swenson, University of Washington; Linda True, TRW Space and Electronics; Bruce Alan Wynn, Collective Technologies

Summary by Carolyn M. Hennings

If there is a hot topic in the SAGE community, it is definitely certification. The issue has been in the air for a number of years, and the SAGE Executive Committee decided it was time to take action toward making a decision to either pursue or drop it. A SAGE Certification Subcommittee was formed, and subsequently a Certification Advisory Council was created.

The purpose of "The Great Certification Debate" was to have a serious discussion about the certification issue. Rob Kolstad asked the panelists to introduce themselves and speak to their positions on the issue. On the pro side of the discussion were Bruce Alan Wynn and Lynda True. The cons were Phil Scarr and Scottie Swenson.

Bruce Alan Wynn expressed his opinion in light of the SAGE charter of advancing systems administration as a profession. The certification project will help with the definition of our system-management standards and then define the requirements for certification. Bruce reminded the audience that SAGE uses the term "guild" in its name, referring to a structure in which more experienced people

help the less experienced. The certification process would provide guidance and direction to individuals who don't know where to start. He concluded by saying that there are a number of different ways to do certification, that some are better than others, and that SAGE needs to do it right.

Phil Scarr reiterated concerns expressed in his *;login:* article ("When Worlds Collide," August 1997). He suggested that the best administrators come from university programs where there is an education focus rather than a certification focus. Experience is a better indicator of ability than certification. Certification is touted as a way to hire but is not effective.

Lynda True explained that her organization has been certifying UNIX administrators for approximately two years. Management recognized inadequate system administration support and lack of training to be potential threats to information security and Internet availability. Although the process has been painful, some benefits have been that hiring has become easier and salaries have risen. She suggested that the peer-review portion of the certification process was an important aspect.

Scottie Swenson commented that certification might be good if done correctly. He commented that most vendor-sponsored certification programs have little value. He expressed concerns regarding the difficulty of managing a certification program. A focus on education should be a priority over certification.

Questions and comments from the audience expressed concerns regarding how the certification process might work. Numerous people brought up the issue of education and how it relates to certification. Suggestions included looking at the certification processes used by the medical, project-management, and aviation professions. An important point brought up by one participant was that if SAGE doesn't certify system-management professionals, some other organization will. If that happens, we run the risk of having to live with something that isn't the best and doesn't quite work.

INVITED TALKS TRACK

Security as Infrastructure

Tom Perrine, San Diego Supercomputer Center

Summary by Kurt Dillard

Tom Perrine convincingly asserted that an effective system administrator must address security at all seven network layers as well as two others, the economic and political layers of your organization. He summarizes effective approaches as "building fences" while everyone else stays busy "shooting rabbits." A long-term solution to effective security will take time to implement, and a few "rabbits" may get in while you build it, but the final result will be much more satisfactory and manageable. Perrine suggests that you undertake implementing a secure infrastructure by first defining goals – figure out what needs to be secure and rank those items by importance. Then decide how you will respond to different types of attacks, automate dealing with unsophisticated attempts, and don't even waste time investigating them personally. Automate dealing with other types of probes and analyze the data personally. Figure out what attacks are so sophisticated that you will have to deal with them personally from start to finish. Basically, this is risk analysis – figuring out what is important, what the threats are, and how much time you need to spend protecting what's important from the different types of hosts. By the way, automate or die, because if you cannot scale your solution it will fail as your network grows.

Perrine then reviewed current attack methods and ways to deal with them. What's being used? Every approach you have ever heard of. Even the weaknesses with known fixes are effective for hacking because most sites never bother to apply the correct fixes properly. The biggest threats are automated exploits that allow attackers to probe numerous hosts very quickly, and any type of authentication that requires a password transmitted as plain text.

To deal with these attacks you need to automate as much as possible in your environment. Implement CfEngine or SMS to allow for quick host installs and simplified patches and upgrades. Set a security policy and automate a method of auditing all hosts to ensure that they are meeting your policy. Segregate the weak hosts from the rest of the network by placing them behind filtering routers. Automate the detection of anomalies and as much as possible automate resolutions for those anomalies. Don't forget to educate everyone who uses your network; "social engineering" is a very effective method for hacking sites. Finally, remember that you have to win the support of your users and your management chain. Build your secure infrastructure slowly so that your users have a chance to get used to it and so that you have time to verify that everything is working at each step.

PRACTICUM TRACK

Teaching Systems Administration

Chair: Lee Damon, QUALCOMM, Inc. Panelists: David Kensiski, Digital Island, Inc.; David Kuncicky, Florida State University; Daniel Klein, USENIX Association; Matt Shibla, Montgomery Blair High School

Summary by Chastity D. Arthur

Consensus at this practicum was that no one has all the answers to teaching system administration, and one particular method is not going to solve this issue. Only a combination of school programs, extension programs, in-house training,

on-the-job training, and vendor courses is the answer for today.

David Kensiski appeared to be in favor of a combination of vendor courses and in-house training. His answers were clear and concise. One audience member asked him what he did with a junior employee who just wasn't grasping the concepts. He politely stated that there was really only one option – either find them something

David Kensiski, David Kuncicky, Daniel Klein, Lee Damon, & Matt Shibla

they can do in another area or terminate them. (This really only brings up another question, what if it's the teaching method?)

David Kuncicky thinks universities need a way of defining the appropriate levels of system administration in order to advance a student, and he compared the teaching methods and coursework with those for programming languages. Kuncicky presented his department's current course design. FSU's program is a mix of books, courses taught on campus by FSU instructors, and third-party trainers. He defined the program by step levels in Systems Administration Proper, SA Tools, SA Networking, and SA Applications. Kuncicky also mentioned that what may separate the different universities is state legislation that mandates the number of hours, which led FSU to a system-administration master's track. He was not necessarily an advocate of the master's track, but FSU's goal was simply to get a program started.

Daniel Klein, tutorial coordinator for USENIX, brought up a major concern of the system-administration industry today: How do you know when or which training is appropriate? He discussed the assumptions that must be made prior to proceeding with any type of training. Klein mentioned that a certain amount of prior experience here and there is needed and that the potential trainee must be self-driven, willing to read, use references, and know when to ask for assistance. Klein was an excellent communicator and he immediately grabbed the audience when he said, "Training does not fix your problems." He followed that up with a concise statement that system administrators can be taught with the see-one, do-one, teach-one theory. He pointed out that indeed vendor training feeds the curriculum, but it is also a general assumption that the instructor knows the course materials but may not necessarily be an expert on the subject. This is what makes USENIX's curriculum flexible; the instructors are the gurus and as technology changes, advances, or grows, so will the course.

The last panelist, Matt Shibla, was there to discuss the Maryland Virtual High School established through Montgomery High School. This is an online curriculum downloaded from Cisco Network Academy to the high school's private network. Shibla stated he felt the program was 60 percent generic and 40 percent vendor-related. The main issue that arose from the audience was that this type of course is limited and teacher-deficient. Shibla pretty much agreed and did not offer any insight into developing the program further other than trying to get teachers certified.

This practicum was not set up to provide the answers but to give the panelists and audience ideas on different directions they could take in designing their own training methods. The audience was primarily focused on the motivation of trainees and teachers and the management issue of "you train them, they leave."

Network Administration and Remote Computing

Moderator: Rob Kolstad. Panelists: Pete Lynch, Jyra Research; Shaula Yemini; Oljad Singh, System Management Arts

Summary by Kurt Dillard

All three speakers presented information about the network-monitoring tools that their companies have developed. Oljad Singh's approach is to focus on critical services and warn of impending and existing problems. His company has installed a server, running their monitoring software, that executes ghost transactions every few minutes on each server being monitored. The monitoring software times each transaction and over time learns what the "normal" performance signature is of each application on each server at different times of the day and different days of the week. When several transactions in a short period of time fall too far outside of the "normal" signature, the monitoring software automatically notifies an operator of a possible problem.

Pete Lynch promoted a product from Jyra that proactively monitors defined business and performance goals. He asserted that service-level agreements don't adequately measure the user's experience,

while Jyra's monitoring solution does so by measuring the response time of applications using a scheduled polling agent to see how each application server is performing. Their scalable solution utilizes distributed Java agents and provides automatic reporting, summaries, and exception alarms.

Shaula Yemini has another proactive real-time management tool, SMARTS, which diagnoses problems before they have had an impact on the network, allowing them to be resolved before users realize that anything has gone awry. SMARTS creates a "codebook" that combines generic models for each network object with the customer's specific topology to create "problem signatures." When the problem signatures are detected the system raises an alert. Her firm already has models for a wide variety of network hardware, and the codebook can be created by connecting their system to some common network-management systems such as OpenView and NetView. The codebook is automatically updated every time the models or topology change.

WORKS-IN-PROGRESS (WIPS)

Coordinator: Peg Schafer, Harvard University

Summary by John Talbot

The WIPs Practicum was more than the advertised "pithy" display of current techniques and issues. It was a nonstop blitz, in concise 10-minute samples, of interesting and thought-provoking descriptions of "real-world" problems and solutions. The WIPs were pitted against one another with a real whip bestowed upon the winner, who was determined by group applause.

Steven Nelson's "Multiplatform Storage Area Network" WIP was a nuts-and-bolts discussion of managing a network of over 1.6 TB of database and data sources in a 24x7 environment. Nike's original fiber-channel network was initially spread out over all storage areas, but problems persisted because of shared data paths for both the backup and data-delivery services. They have used the EMC array and parallel pipes to have transparent access to data dumps without interfering with network and system resources. They still have large-size backups and multiple filesystems for data storage. They are looking into ways of using the Veritas volume manager filesystems on their EMC array to have a commonly mountable local vxvm filesystem between heterogeneous platforms.

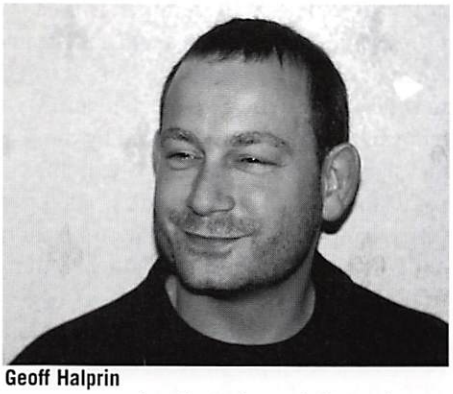
John Buckman <john@lyris.com> presented a unique implementation of email as a mission-critical application. Instead of a centralized MTA and MDA, Lyris opted for a thin multithreaded mail daemon that uses SQL as the message store. This enables his site to utilize the text searching and user access of the existing SQL structures to manage email. Also, they are able to deliver an email interface using HTML and Tcl more easily with the existing SQL structures.

Lowell Snyder <lsnyder@ptc.com> presented another good email implementation. (I might be a little biased since I was a cohort of this WIP, but it did come in a close second.) Lowell presented some of the work done at Parametric Technology to remove the standard UNIX aliasing from the central MTA and institute LDAP hooks into the internal mail exchangers to directly deliver mail based on the company's managed LDAP database. A motive was that a majority of users were consolidated on a central POP/IMAP MDA, and the management of simple UNIX mail aliases became cumbersome as the user base grew to over 4,000. Snyder described the code changes required on the sendmail and LDAP daemons necessary to invoke the features and handle several exceptions for whitespace and parsing symbols not generally understood or interpreted by general sendmail parsing rules.

Tom Limoncelli <tal@lucent.com> presented "Tricks you can do when your firewall is a bridge." This WIP stemmed from a project Limoncelli had at Bell Labs/Lucent when cutting over backbone routers to new firewalls to the Internet. The problem was originally complicated by the fact that he didn't have access to all the routers in the schema and still needed to implement the firewall changes transparently. Of note was the fact that when the firewall acts as a bridge and doesn't have an IP number, it is less likely to be subject to hacking attempts since it has no TCP/IP-bound interface. See <http://www.bell-labs.com/usr/tal>.

Charles Tatum <tatum@nswc.navy.mil> of the U.S. Navy presented "Computationally Expensive Intrusion Prevention." He focused mainly on his modifications to the popular Crack program. He surmised that a massive amount of time was being used by the Crack code to test less probable circumstances of password guessing. His new approach was to apply common rules first. For example, he found that most people don't use more than one word for their password, so applying a number of permutation rules to the password guessing before trying "obvious" guesses makes running Crack computationally expensive. Also, implementing a separate dictionary of names ahead of the standard dictionary will weed out more obvious guesses. Reducing the salt guesses also lowers process runtime. Tatum's approach manipulated the original Crack encrypt and compare looping structure from Dict→Rules→Users→Salts to Rules→Dict→Users→Salts and implemented his common-guess strategies to greatly lower computing times.

Geoff Halprin <geoff.halprin@sysadmin.com.au> of SysAdmin Group, winner (by a hair over Lowell Snyder) of the WIP prize, gave a stunning talk about the "Taxonomy of Best Practice." This was one of the clearest talks classifying and describing what sysadmins do that I have heard in a long while. Halprin not only explored the need for sysadmins to understand their


Geoff Halprin

own personal technical capabilities, but also explained that the user community needs to understand what sysadmins do and of what they are capable so that sysadmins can be properly matched to the environments they support. He defined system-administration core competencies as control, organization, protection, optimization, and planning, and he included a diagrammatic breakdown of each of these characteristics. He established models for organizing these responsibilities on a set of five levels, with three to five of those levels being consistent across all platforms. The five levels are determined by a Capability Maturity Model by which certification is based upon core competencies. This WIP would be a great basis for a future LISA invited talk.

Michael Ewan <michael.ewan@tek.com> of Tektronics described using LDAP to create printer definitions and determine printers by class (e.g., color, resolution, paper size) and location. He described how the current workstation environment DISPLAY variable and user profiles could be used to reference the "nearest" user-default printer. He has also looked into ways to use the LDAP information base to manage printers. His environment eased implementation by its homogeneous use of Tektronics printers and the standardization of UNIX workstations.

Andrew Hume <andrew@research.att.com> and **Tom Scola** from AT&T Labs presented "How to Handle Microsoft Attachments" in UNIX email. They wanted to have a UNIX-based reader that would be able to read email attachments in realtime. To reduce the need for specialized windowing software, they devised a plan that sends the Microsoft attachment to an NT system that runs a PostScript converter and sends it back to the Xwindows-based MTA. Unfortunately, time was running short at this point of the WIP session and many of the details were abbreviated before Hume could complete his WIP.

BOFS

Summary by Douglas Stewart

Variable Length Subnet Masks on TCP/IP Networks

Mike Andrews

The basics of Class A/B/C and their corresponding netmasks, and reserved classes for testing and internal use, were covered. Using all of your addresses in a single network is wasteful, especially if your company has a Class A address allocated to it and has its network broken up into geographically separate chunks. By modifying your netmask, you can break up your network into smaller, separate pieces that can be routed separately. A useful example is an ISP that resells large number of T1s. Typically you'll have a subnet composed of only the router on each side. By setting your netmask to 255.255.255.252, you break your network into 64 subnetworks with four addresses, which include a network address, the broadcast address, and two usable host addresses you can use for the routers. Routing protocols and commands for troubleshooting routing problems were suggested. Some light reading (RFCs 950, 1918, and 1878) was suggested.

BSDI

Doug Urner

Most of this presentation was on new features of BSDI 4.0. Filesystem code updates: soft update (delays certain file operations for speedup, very temporary files may never be created); even out update; 64-bit file offsets; mount options to disable access time updates; mount options for sync/asynch writes; and soft read-only. Networking improvements: PCB lookup hashing; IP address hashing for fast virtual hosting; per-address IP statistics; and kernel-level packet filter. IPv4 enhancements: slow start; congestion avoidance; multi-cast; large windows; MTU discovery; and IPSEC. IPv6 support! Includes SAMBA, IP/IPX, Novell 3.x file and print services, VPN support. NFS: v2 and v3 support over UDP or TCP; NFS lock daemon. New network media: frame relay. SMP: performance improvements (user-level processes show best improvements); threads are all user-level for now but the kernel will be threaded in the future. New hardware improvements include bootable CD-ROMs and plug-and-play Ethernet, modem, and sound cards. Binaries are now ELF, and the math libraries have been proved. There's a console debugger and trace facility called Kdebug and KTR. Things to look for in the future: Linux binary compatibility, Java application environment, finer-grained SMP with kernel threads, SPARC port, ATM, and channelized T1 and T3 support.

AFS

Esther Filderman

The people who attended this BOF were almost entirely AFS users from university environments. The first topic was release dates for Linux and NT-based AFS servers – apparently in February 1999. Transarc, which has been bought by IBM, was a topic for heavy discussion. It has a new CEO, has changed its mind about dropping the development of AFS, and has opened a London office. Something else people were looking for was Kerberos 5 support. There were complaints about poor support from Transarc, especially with the 5 pm EST closing time that is inconvenient for West Coast customers. KNFS was discussed as something that

people were experimenting with and had had some success with. In the end, people had little faith in most of the alternatives to AFS (NFS, DFS, CacheFS) and felt that the advantages of AFS outweighed the problems they had encountered.

ADVANCED TOPICS WORKSHOP

Adam Moskowitz, Facilitator
Rob Kolstad, Co-chair and Scribe

Summary by Bruce Alan Wynn

We first went around the room introducing ourselves, the quantity of users and the quantity and type of hosts we supported (whether individually or as part of a team), and two to five topics we wanted to discuss during the day. We came up with ranges of up to 10,000 users, 10,000 PCs, 2,000 Macs, and 3,000 UNIX hosts of various flavors. Other notes were multiple terabytes of disk storage (with projected short-term growth to exceed a petabyte), strange printer requirements, and extremely high growth rates (up to 400% a year).

We determined that we wanted to talk, in general, about:

- Consistency/standardization in sysadmin practices as organizations grow
- Cool system-administration tools and paradigms
- Specific hot technologies/paradigms to prepare for/crystal ball

First, we discussed the issue of internal consistency and standardization in technical practices. We tried to look at the "problem," but realized that we all had slightly different ways of looking at it — which was not surprising considering that we had 31 people in the room, all with different backgrounds and experiences. We seemed to agree in general that creating standards is challenging, enforcing them is a hard problem, and that there are many more variables than may be obvious at first look. The concept of a "taxonomy" or categorization of problems into areas seemed to make sense to a lot of the folks present.

We next had a free-form discussion on cool system-administration tools and paradigms. Some general comments were:

- DSL is great. (10 people have 56K or more to the home, all but one of those has 56K or faster bidirectionally; six people have >128K. Most of these are business-paid and not individual-paid.)
- MRTG (discussed in Tobias Oetiker's paper at the conference) was hailed as a wonderful network-mapping tool. It uses SNMP polling on a 5-minute interval and creates Web pages with usage graphs. It ages data appropriately and is freely available. Big Brother, a systems monitoring package, integrates with MRTG and is also freely available.
- Intrusion-detection systems now are in the same sort of not-yet-well-understood position as firewalls were a decade ago. While then we had free firewalls which later became commercial, now we have commercial intrusion-detection systems (IDS) even though the problem is neither understood nor solved.
- Turnover can be interesting. 16 people changed jobs at least once in the past year. Four of these were internal (same company) job changes. Raises in the new job ranged from 0-90% and seemed to average around 28%. And 14 people present have open requisitions they are actively hiring for.
- Enforcing the use of a PDA like the PalmPilot has improved the follow-through for members of the group. Many folks at one company have bought one with their own money.
- 19 of those present carry a cell phone; 26 carry pagers; seven carry authentication devices. A few have two-way pagers; 12 participants pay at least part of their monthly fees for the portable communications devices.
- Five members use a Ricochet or similar device for wireless digital communications. 14 more would use it if it were available in their area.
- Seven people have an agreement to attend conferences annually. In spite of that small number, 10 have some kind of permission to attend more than two per year. A couple can go to even more if they have papers presented at them. Everyone pretty much gets at least one per year. About half can attend two or more per year, depending on circumstances.
- Some cool utilities are ssh and Curl. LDAP or similar directory services are on the rise; 11 attendees have this.
- Cordless phones in machine rooms are a major win.
- Tools sometimes die for lack of nurturing. It would be nice if there were some way to solve that problem (like a MacArthur grant type of thing). Even finding current versions is too hard. See <ftp.sage-au.org.au> for lots of sysadmin tools.

Next we discussed hot technologies, rumors, and similar prognostications. One hot technology we talked about is XML, the Extensible Markup Language. It is self-verifying, easy to parse, easy to search, and has a universal file format. It's different from SGML in that it doesn't include the hard-to-implement features. XML supports Unicode. Unicode is the next hot technology we see on the horizon. It represents all characters (including nonRoman alphabets like Cyrillic, Hebrew, and Farsi). Microsoft Office 2000 uses Unicode; rumor has it that Word already supports it.

Other predictions are: Voice over IP will be a hot technology soon; directory services are becoming more important; applications will support more location independence; voice input and/or recog-

nition will grow in the next year; and digital camera use will continue to rise.

SAGE COMMUNITY MEETING AND CANDIDATES FORUM

Summary by Carolyn M. Hennings

The annual SAGE Community Meeting had a significantly larger attendance this year than in the past two years that I've attended. Pat Wilson kicked off the meeting with announcements, introduced the candidates for the upcoming Executive Committee elections, and moderated a question-and-answer session.

Announcements included the status of publications in the "Short Topics" series. *Educating and Training Systems Administrators* is in the mail to SAGE members. Pat provided an update on the certification debate, saying that an advisory committee has been formed and charged with the investigation of whether or not SAGE should continue to pursue the certification issue and how it should be done if SAGE chooses to move forward. Pat mentioned the efforts of the "Day-in-the-Life" survey and announced that Rob Ferrell and Brian Kirouac have been named SAGE Historians. Items to watch for are "How-To Notes," revamped Rosetta Stone, new booklets on site audits and on hiring system administrators. Efforts are under way in reviewing the ethics policy and in developing a mentoring program.

Moving on to the candidates forum, Pat asked the candidates to introduce themselves, and the floor was opened up for questions. Question topics included the certification issue, the ethics policy, education and training, the inclusion of Windows NT topics in the LISA program, and the market's view of SAGE.

In the BPF format, the SAGE Community Meeting continued under the guidance of Hal Miller, SAGE Executive Committee President. The open-forum discussion focused on the professional development of systems administrators in the sense of gaining skills to communicate and work with management. David Parter, chair for LISA '99, took many suggestions for topics for the next conference. The smaller group meeting provided an informal opportunity to meet other individuals who are actively involved with SAGE at the national and local levels.

CLOSING SESSION

Summary by Carolyn M. Hennings

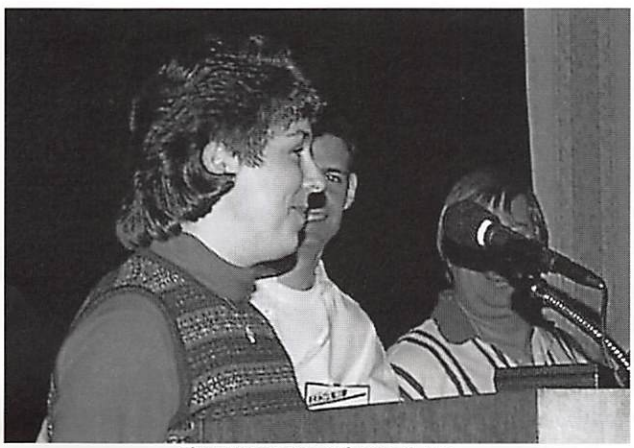
At a lot of conferences the attendee population markedly diminishes on the last day of the conference. LISA is different for one reason. As always, Rob Kolstad's LISA Quiz Show is a major attraction at the conclusion of the conference.

This year was no exception. Maintaining the same format from previous years and always making technical improvements, the Quiz Show gets better and better. Conference attendees vied to answer questions in areas such as UNIX administration, the WWW, computer executives, physics, circus acts, electrical current, coins, certification questions, and European dictators. In the category of conductors questions about electrical conductors set the stage, but the final question was "Who wrote *West Side Story*?" One never knows what to expect.

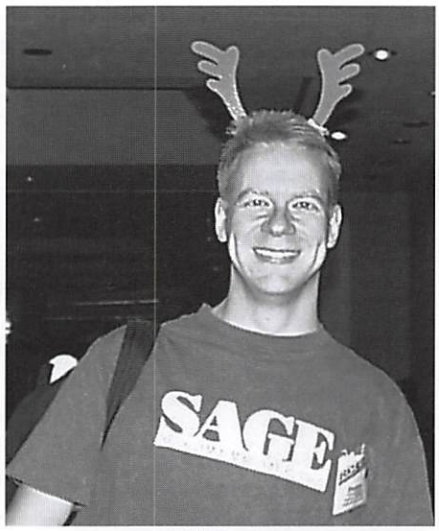
A bonus this year was the "Tournament of Champions." Last year Snoopy Beagle, who hails from Germany, lost to Hal Pomeranz. Snoopy then challenged the fairness of the Quiz Show, saying that too many questions were based on American pop culture. Apparently Rob Kolstad heard enough feedback over the past year regarding this issue and decided to rectify the situation.

The "Tournament of Champions" was a contest between this year's winner, last year's winner, and Snoopy. Notable categories for the final round included European history, television shows, security, and match the dictator. In a surprising victory, Daniel Boyd made mincemeat of his competitors. Who knows what will happen next year!

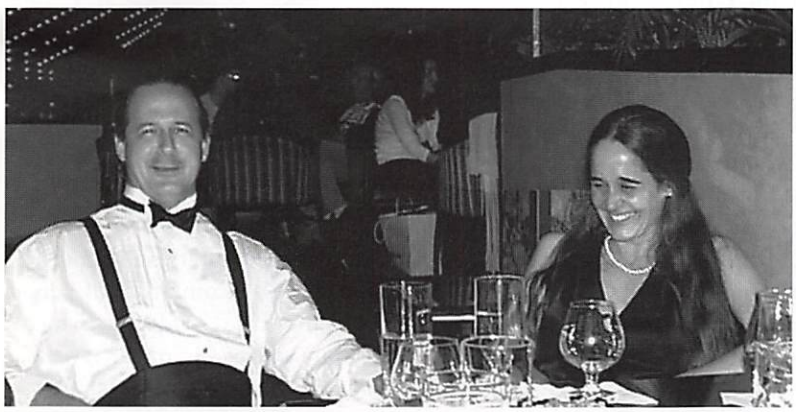
The 1998 Champion, Daniel Boyd, with Rob Kolstad



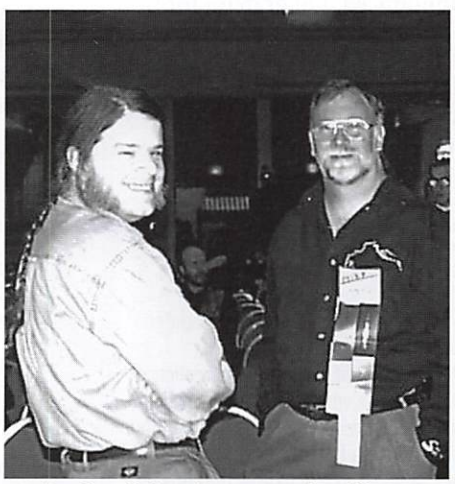
Tina Darmohray, Winner of the
1998 SAGE Outstanding Achievement Award



Snoopy



Adam Moskowitz & friend, demonstrating the latest LISA dress code



Dan Geer & Greg Rose comparing ribbons

Enjoyed the photographs? Attending a USENIX conference? Want to contribute to others' enjoyment? If you take good pictures at the conference, send them to us. If we can use them in *;login:*, though we can't promise you fame or wealth, we do promise to credit you as photographer.

SAGEnews & features

Off to See the Wizards

by Tina Darmohray

Tina Darmohray, editor of SAGE News & Features, is a consultant in the area of Internet firewalls and network connections, and frequently gives tutorials on those subjects. She was a founding member of SAGE.

<tmd@usenix.org>

I'm just back from the very successful and well-attended LISA conference, where the brave attendees ("*Boston* in *December*?") were rewarded with downright balmy weather. There was lots to see and do, and, most of all, fellow admins to talk to; I always learn so much in the short time I spend at LISA. The high point of the conference for me was receiving the SAGE Outstanding Achievement Award. Fortunately I wore a red turtleneck sweater that morning, which helped tone down the matching cheek-color I was sporting as I heard my name announced. I'm grateful for the honor.

I think recipients of awards like this one seldom come to the podium alone. By that I mean that they've likely been assisted, influenced, mentored, and encouraged by a lot of people along the way, and therefore represent a whole group of people's effort. I know that's the case with me. Many of the people I worked with early in my career were very willing to answer questions and point me in the direction of additional sources of information.

Later, when my coworkers and other professional friends felt I had useful information to share, they encouraged me to do so by submitting a paper to LISA, then editing the *Job Descriptions* booklet, and now editing for *;login:*. Each time I considered doing any of these I always felt ill-qualified. Each time, however, I also met willing mentors who shepherded me along until I became able to accomplish the task on my own.

I liken my experience to Dorothy and the Wizard of Oz. She very much wanted an audience with the Wizard, anticipating how much he could help her. At first, the Wizard seemed unapproachable, and Dorothy questioned her qualifications and convictions. Again her supportive compatriots, who were pivotal during her journey to the Emerald City, continued to urge her forward, even though the outward image of the Wizard was so daunting. Luckily, she found that the Wizard wasn't really scary and that he'd also help her toward her goal.

I'd like to encourage all of you to start your own journey toward Oz! One way to get going is to write an article for *;login:*. If you've got a neat tool, interesting solution, or great idea to share with the SAGE community, we'd love to hear from you. Just in case the unknown twists of the yellow brick road are what's keeping you from getting started, let me share some answers to the most FAQs I get from first-time *;login:* contributors.

Format: Since we do the formatting, all we need are ASCII submissions.

Length: Theoretically, there's no minimum or maximum length for an article. Something chapter-length might be a little long, but we can always work with you to split it across more than one issue. (Hey, these are good problems to have!)

Deadlines: Our publishing deadline rolls around every other month, usually during the first week (check the back cover of a recent issue), but you can submit something at any time. If you don't know if your topic would be suitable, you can run it past us ahead of time and we'll let you know up front, before you've sunk a lot of effort into polishing your prose.

Help: Most important, if you've got the topic, but share the common, self-perceived, "I can't write" syndrome (which I've certainly ascribed to myself), we'll help you with that. We have bona fide

SAGE, the System Administrators Guild, is a Special Technical Group within USENIX. It is organized to advance the status of computer system administration as a profession, establish standards of professional excellence and recognize those who attain them, develop guidelines for improving the technical and managerial capabilities of members of the profession, and promote activities that advance the state of the art or the community.

All system administrators benefit from the advancement and growing credibility of the profession. Joining SAGE allows individuals and organizations to contribute to the community of system administrators and the professions as a whole.

SAGE membership includes USENIX membership. SAGE members receive all USENIX member benefits plus others exclusive to SAGE.

SAGE members save when registering for USENIX conferences and conferences co-sponsored by SAGE.

SAGE publishes a series of practical booklets. SAGE members receive a free copy of each booklet published during their membership term.

SAGE sponsors an annual survey of sysadmin salaries collated with job responsibilities. Results are available to members online.

The SAGE Web site offers a members-only Jobs-Offered and Positions-Sought Job Center.

wizards of editing on staff for just that reason. And you should know that *everything* in *;login:* gets edited to some degree, so you really, really are not going to stand out, or have any reason to be embarrassed, if your article needs some editorial assistance. In fact, consider any editing that is done as free training on how to improve your writing style. I can state from experience that the editing process has been the single most effective tool in bettering my own writing skills.

Topics: Finally, if you like to write but are in search of a topic, we brainstormed on some of the neat topics and additional items we'd like to feature in *;login:* in the upcoming year. They include:

Webmastering

security

NT integration

64-bit programming

open source

production (email) servers

most interesting Web sites

most useful tools

USENIX community news

If you are working on any of these or are interested in writing about them, please get in touch with us.

So, if you're inclined, get started on contributing to *;login:*. I can promise you'll get mentoring and encouragement, and you'll enjoy the wizards you meet in the process!

Extending Our Goal Set

by Hal Miller

Hal Miller is president of the SAGE STG Executive Committee.

<halm@usenix.org>

As I complete the intrusion-recovery checklist for a series of break-ins, I sit here contemplating next steps for SAGE. I am now at a university, in an environment very different from those I've been in for most of my career. In the past I have "owned" the network, servers, and clients. If any user had root access (or equivalent), the machine was on a separate firewalled segment, or I worked closely enough to trust the individual. Now, I have no access to the wiring closets nor their contents, except as a general user of network services. I have control over only a small number of machines, and access of any kind (often excluding root) to only a small percentage of the machines on my net. There are no promulgated policies; my docs are still sitting on people's desks for review.

Three machines were compromised this week, all by the same attack method, although by two unrelated groups of attackers. I had no access to any of those machines, and in fact wasn't even aware of the existence of one of them (installed on the network by the owner and the university's networking support group). The attackers in both cases also tried machines my group does control, all of which fended off the attacks and reported to us. We observed many other attacks, but these used "old" methods and failed to gain access to those machines logging the attempts.

Suddenly we were heroes. We went in and collected legal evidence, proving to the machine owners the extent of damage done, then rebuilt the boxes per industry standard. We now have "shared" control, which probably means the users do whatever they desire, and we get the blame when that causes a problem.

What exactly are the issues here, and how can SAGE help?

- How do we get policies implemented?
- What do we do if we can't get policies implemented, or in the interim while review is under way?
- How should we deal with responsibilities split between system and network admins, especially when they don't agree on requirements?

- How should we deal with responsibilities split between system administrators and user-owners who have root/administrator access?
- How do we deal with requirements for sharing facilities/services on our own networks to untrusted boxes, let alone remote nets?

I have no immediate authoritative and conclusive answers. I do think, however, that SAGE is the right place to establish those conclusive answers. Many of us are faced with similar situations, and these problems affect the industry as a whole. They are among the kinds of problems SAGE should tackle.

How? Well, thus far SAGE has been concentrating on bootstrapping itself. Current emphasis is on the advancement of the individual sysadmin through creation and operation of conferences and workshops, ethics, certification and educational projects. It is time to start looking at what we can do for the industry in addition to supplying qualified sysadmins. The Short Topics booklet, *System Security: A Management Perspective*, is a start in this arena, but we need to plan out a strategy for far more and bigger goals. What might those goals be? Some possibilities:

- Raising the security standard of the average site so high that "casual" cracking ceases to be of interest
- Raising the consciousness level of the average site's management regarding the requirement for an appropriate level of applied system administration resource and security effort
- Creating a trusted place that organizations or individual sysadmins might turn for immediate assistance for any form of sysadmin problem, and making it known and accepted
- Establishing minimum "standards of practice" guidelines – a tricky and controversial area

We have made some progress already, such as Short Topics booklet *A Guide to Developing Computing Policy Documents*, but we are still in the very early stages of addressing the scope of problems faced by the "average" sysadmin. Our approach so far has been to provide an overview of the methods we might apply to the "black art" of system administration, and to assume that local sysadmins will know how to follow through. Perhaps most of us old-timers can do this, perhaps not. Can newer folks do so, or are they in need of more concrete examples? The new "How-To Notes" series in *;login:* should address this to some degree, but is that enough? It seems that uncountable sites out there have site sysadmins who are not SAGE members. If they don't take our soon-to-be education programs, will they be needing checklists? Is it right that we provide such a thing? I'd like to see some discussion on these questions. Don't hesitate to write for *;login:*, to <sage-members>, or directly to me.

By the time you read this, another election will be behind us, and a new Executive Committee will be setting new directions. While they do so, it is not too early to begin thinking about what you might wish to do to prepare for the committee that will follow them and how to get those projects started. It might even be too late in some cases, Other organizations are attempting to make, codify, or otherwise influence our profession from their own viewpoints, typically profit-centered rather than sysadmin-centered. We can't drop the ball for a moment, especially at this point in our development.

how-to

Install Anonymous FTP

This note describes how to configure an anonymous FTP server on a UNIX-like operating system. The steps for configuring the freely available Washington University FTP daemon software (WU FTP) are described, but administrators wishing to use their vendor's anonymous FTP server may do so simply by skipping Section 1, Section 2.7, Section 3.1 (though see Section 3.1.2 for guidelines on how to create an /etc/ftpusers file), and Section 3.3.

by Hal Pomeranz
Hal Pomeranz is the Chief Operations Architect for Corio, Inc., an Application Services Provider based in Redwood City, CA.

<hal@deer-run.com>

The WU FTP software should be used if the administrator wishes to allow file uploads in a secure fashion (Section 2.7). In particular, a good set of requirements for file upload areas includes

I. No user may get a directory listing in the upload area or any of its subdirectories (helps prevent disclosure of proprietary information in file or directory names).

II. Files may only be uploaded into subdirectories of the upload area (helps protect `~ftp/{bin,lib,dev,etc,pub}`).

III. Files that have been uploaded may not be downloaded by any anonymous user (prevents your site from being used as a "warez" site for the dissemination of copyrighted materials).

IV. Files that have been uploaded may not be overwritten (helps prevent confusion and trojan-horse attacks).

Requirement I plays havoc with graphically oriented, "point-and-click" FTP clients that insist on getting directory listings in order to function properly. Most of these clients allow the user to input pathnames from the keyboard, but some amount of user education is likely to be required.

Requirements III and IV are enforced by configuring the WU FTP daemon to make all uploaded files be read-only (mode 400) owned by root. This implies an administrator or some external process that retrieves files from the upload area and makes them available to internal users.

1. WU-ftpd

The Washington University FTP daemon is the de facto standard FTP server for anonymous FTP sites. This is primarily due to the wide variety of features it supports – features that make the code more difficult to audit and therefore more dangerous from a security perspective. If you only need to allow "vanilla" anonymous FTP access (no file uploads, no nonanonymous access), then skip this section and use your OS vendor's ftpd.

1.1. Download

1.1.1 Connect to `ftp://ftp.academ.com/pub/wu-ftpd/private/`

1.1.2 Download latest beta release (currently `wu-ftpd-2.4.2-beta-18.tar.Z`)

1.1.3 Unpack in appropriate source directory

```
# mv wu-ftpd-2.4.2-beta-<vers>.tar.Z <srcdir>
# cd <srcdir>
# zcat wu-ftpd-2.4.2-beta-<vers>.tar.Z | tar xf -
```

where `<vers>` is the appropriate version number (see previous step) and `<srcdir>` is

ANNOUNCING THE SAGE "HOW-TO NOTES" SERIES

With the accompanying first entry by Hal Pomeranz, SAGE is pleased to announce the commencement of the How-To Notes series. Based on an idea floated in last year's <sage-member> survey, this series is a collection of basic information intended to assist sysadmins in quickly getting something running. They will cover such points as where to obtain software, what hardware may be required, and what steps are necessary to get a minimal service configured for "average use." The notes are not designed for advanced configurations, complex installations, or specialized situations. All will eventually be available from the SAGE Web pages. The series editor (not yet selected) will manage new topics and a review cycle.

Questions, comments, and suggestions are always welcome. We hope this series proves valuable to the membership.

The most important and most difficult part of setting up any anonymous FTP installation is getting the directory permissions right. Errors in this step can allow outsiders to modify or overwrite files in your anonymous FTP area and possibly gain shell access to your machine.

some directory on your system where you keep third-party source code.

1.2. Build Process

1.2.1 Move to configuration directory.

```
# cd wu-ftpd-2.4.2-beta-<vers>/src/config
```

1.2.2 Edit appropriate configuration file for your OS (consult `../../INSTALL` for OS type information). If `USE_ETC` is set, replace this with `USE_ETC_FTPD`.

1.2.3 Return to top-level source directory.

```
# cd ../..
```

1.2.4 Initiate build process.

```
# sh build CC=<comp> <target>
```

where `<comp>` is your C compiler (cc by default) and `<target>` is the appropriate OS identifier.

1.3. Install

1.3.1 Edit top-level Makefile and set `BINDIR`, `ETCDIR`, and `MANDIR` appropriately. WARNING! On most architectures, the default values for these variables will overwrite existing OS binaries and other files. This is probably *not* what you want.

1.3.2 Trigger install process.

```
# make install
```

2. Create anonymous FTP directory

When a user logs in as anonymous, FTP daemons call `chroot()` to restrict that user's access only to files in your anonymous FTP area. Make sure that your anonymous FTP area lives in its own partition and don't make any symbolic links into or out of this area. Also note that the most important and most difficult part of setting up any anonymous FTP installation is getting the directory permissions right. Errors in this step can allow outsiders to modify or overwrite files in your anonymous FTP area and possibly gain shell access to your machine.

2.1. Create top-level FTP directory

2.1.1 Make directory.

```
# mkdir <path>
```

where `<path>` is where the anonymous FTP area is rooted (e.g., `/usr/local/ftp`, `/export/ftp`, etc.).

NOTE: This step may not be required if the anonymous FTP area is going to be located on an already-mounted partition, or you may have to modify your fstab file and mount the FTP area by hand before proceeding.

2.1.2 Set ownership and permissions.

```
# chmod 755 <path>
# chown root <path>
# chgrp root <path>
```

dribble volume manager configuration

c1tXdY (get X,Y below)

0,0 ROOTDISK	2,0 ROOTMIRROR	4,0 SPARE
0,1 usrlocal/local_raidlog	2,1 elib/elib_raidlog	4,1 depot/depot01
0,2 elib/elib01	2,2 elib/elib03	4,2 elib/elib05
0,3	2,3	4,3
0,4	2,4	4,4
1,0 usrlocal/local01	3,0 usrlocal/local02	5,0 usrlocal/local03
1,1 elib/elib07	3,1 elib/elib08	5,1 elib/elib09
1,2 elib/elib02	3,2 elib/elib04	5,2 elib/elib06
1,3	3,3	5,3
1,4	3,4	5,4

RAID-5 volumes: usrlocal, elib (these are not mirrored but do have RAID5 parity info)
Simple volumes: depot (not mirrored, no parity info!)

Boot device:
/io-unit@f,e0200000/sbi@0,0/SUNW,soc@2,0/SUNW,pln@a0000000,78d343/SUNW,ssd@0,0:a

Boot mirror:
/io-unit@f,e0200000/sbi@0,0/SUNW,soc@2,0/SUNW,pln@a0000000,78d343/SUNW,ssd@2,0:a

2.1.3 Move to new directory.

```
# cd <path>
```

2.2. Install `ls` *program.*

2.2.1 Make `~ftp/bin` directory.

```
# mkdir bin
```

2.2.2 Copy binary.

```
# cp /bin/ls bin
```

2.2.3 Set ownership and permissions.

```
# chmod 111 bin bin/ls
# chown root bin bin/ls
# chgrp root bin bin/ls
```

2.3 Install shared libraries.

NOTE: Commands below are appropriate for Solaris systems. Consult OS documentation to find out appropriate libraries for other systems. Skip this step if you can build a statically linked ls program on your machine (ls sources available from any GNU archive in the fileutils package).

2.3.1 Make `~ftp/lib` directory.

```
# mkdir -p usr/lib
```

2.3.2 Copy libraries.

```
# for lib in ld.so.1 libc.so.1 libdl.so.1 libintl.so.1 \
>           libw.so.1 nss_files.so.1
> do
>     cp /usr/lib/$lib usr/lib/$lib
>     chmod 555 usr/lib/$lib
> done
```

2.3.3 Set ownership and permissions.

```
# chmod 111 usr usr/lib
# chown root usr usr/lib usr/lib/*
# chgrp root usr usr/lib usr/lib/*
```

2.4. Make devices.

NOTE: The arguments for `mknod` shown here are for Solaris systems. For other machines, copy the major/minor device numbers, ownership, and permissions from `/dev` (the `ls -l` command may be used to show major/minor device numbers).

2.4.1 Create `~ftp/dev`.

```
# mkdir dev
```

2.4.2 Create device files.

```
# mknod dev/tcp c 11 42
# mknod dev/zero c 13 12
```

2.4.3 Set ownership and permissions.

```
# chmod 111 dev
# chmod 666 dev/tcp
```

Make sure that all of the files and subdirectories in the ~ftp/pub directory are owned by root or somebody other than the anonymous user, ftp, so that there is no chance of the files being overwritten.

```
# chmod 444 dev/zero
# chown root dev dev/tcp dev/zero
# chgrp sys dev dev/tcp dev/zero
```

NOTE: The permissions for dev/tcp shown above are correct.

2.5. Configure ~ftp/etc directory.

2.5.1 Create ~ftp/etc directory.

```
# mkdir etc
```

2.5.2 Create ~ftp/etc/passwd file as follows:

```
root:x:0:1:0000-Admin(0000):/:/sbin/sh
ftp:x:90:1:Anon FTP User:/:/sbin/sh
```

WARNING! *Do not* put actual passwords in the ~ftp/etc/passwd file! *Do not* create ~ftp/etc/shadow or similar file!

2.5.3 Create ~ftp/etc/group file.

```
root:*:0:
other::1:
```

2.5.4 Create ~ftp/etc/nsswitch.conf file (not required on nonSolaris machines).

```
passwd: files
group: files
```

2.5.5 Create the welcome.msg, message.toomany, and pathmsg files in ~ftp/etc as documented in Appendices B-D.

2.5.6 Set ownership and permissions.

```
# chmod 444 etc/*
# chmod 111 etc
# chown root etc etc/*
# chgrp root etc etc/*
```

2.6. Make download area.

NOTE: Files in this area will be readable by anonymous users. Make sure that all of the files and subdirectories in this directory are owned by root or somebody other than the anonymous user, ftp, so that there is no chance of the files being overwritten.

2.6.1 Make ~ftp/pub directory.

```
# mkdir pub
```

2.6.2 Set ownership and permissions.

```
# chmod 555 pub
# chown root pub
# chgrp root pub
```

2.7. Make upload directories.

NOTE: Skip this section if you do not wish to allow uploads to your server at all. Do not allow uploads if not using WU FTP.

2.7.1 Make ~ftp/incoming directory.

```
# mkdir incoming
```

2.7.2 Set ownership and permissions.

```
# chmod 111 incoming
# chown root incoming
# chgrp root incoming
```

2.7.3 Create one or more individual upload areas (substitute some directory name for <dir> below).

```
# mkdir incoming/<dir>
# chmod 311 incoming/<dir>
# chown <uid> incoming/<dir>
# chgrp root incoming/<dir>
```

where <uid> is the numeric UID you will use for the ftp user in the system password file.

3. Other system-configuration tasks

With the anonymous FTP area successfully configured, the WU FTP software itself must be configured, password entries created, and modifications made to inetd.conf.

3.1. Create WU FTP configuration files.

NOTE: Skip this section if not using WU FTP, but do create a file called /etc/ftpusers per Section 3.1.2 (but *not* /etc/ftpd/ftpusers as documented there).

3.1.1 Make /etc/ftpd directory

```
# mkdir /etc/ftpd
# chmod 700 /etc/ftpd
# chown root /etc/ftpd
# chgrp root /etc/ftpd
```

3.1.2 Create /etc/ftpd/ftpusers file. This file contains the list of all users (one per line) *not* allowed to FTP into the server. It should include root, daemon, bin, nobody, and similar accounts which are not associated with real users.

3.1.3 Make symlink from traditional /etc/ftpusers location.

```
# ln -s /etc/ftpd/ftpusers /etc/ftpusers
```

NOTE: If /etc/ftpusers already exists, make sure that /etc/ftpd/ftpusers is a superset of this file, then remove /etc/ftpusers and create the symlink.

3.1.4 Make empty ftpconversions file.

```
# touch /etc/ftpd/ftpconversions
```

3.1.5 Create ftpaccess configuration file from Appendix A.

3.1.6 Set ownership and permissions.

```
# chmod 600 /etc/ftpd/ftp*
# chown root /etc/ftpd/ftp*
# chgrp root /etc/ftpd/ftp*
```

3.2. Make password entry.

3.2.1 Create entry for the ftp user in /etc/passwd.

```
ftp:x:<uid>:<gid>:Anonymous FTP Account:<path>:/dev/null
```

Create /etc/ftpd/ftpusers file. This file contains the list of all users (one per line) not allowed to FTP into the server. It should include root, daemon, bin, nobody, and similar accounts which are not associated with real users.

Do not *use a valid DES password string for the* `ftp` *user. Instead use a string like* `*` *or* `LOCK` *or* `*NP*`*. Do* not *use an empty password field for the* `ftp` *user!*

<uid> is the numeric user ID used in Section 2.7.3 and <gid> is some unused numeric group ID. <path> is the root of the anonymous FTP area as used in Section 2.1.

3.2.2 Create corresponding entry in `/etc/shadow` or other file as required by your operating system. *Do not* use a valid DES password string for the `ftp` user. Instead use a string like `*` or `LOCK` or `*NP*`. *Do not* use an empty password field for the `ftp` user!

3.3. Make `inetd` *changes.*

NOTE: Skip this section if not using WU FTP.

3.3.1 Edit `/etc/inetd.conf` (on some systems, `/etc/inet/inetd.conf`) and replace the existing entry for `ftp` with `ftp stream tcp nowait root <BINDIR>/ftpd ftpd` where <BINDIR> is the directory chosen for the install of WU FTP binary in Section 1.3.1.

3.3.2 Get process ID of running `inetd` using `ps` and `grep`.

3.3.3 Send SIGHUP to running `inetd`

```
# kill -HUP <pid>
```

where <pid> is the process ID determined in the previous section.

Appendix A. WU FTP `ftpaccess` Configuration File

```
# real users may log in from the 172.16.0.0/16 network— you
# probably want to use your internal network address space here
# instead. Anonymous users may log in from any IP address.
#
class users real 172.16.0.0 localhost
class anon anonymous *
# Three login attempts are permitted before the user is dropped and
# a message is logged via syslog.
#
loginfails 3
# Only ten real users are allowed at any time. 100 anonymous users
# are allowed any Saturday or Sunday, or any weekday between 6pm
# and 6am local time. At other times, only 60 anonymous users are
# allowed.
#
# Obviously, you'll need to tune these parameters for your server
# and bandwidth usage limits...
#
limit users 10 Any /etc/msg.toomany
limit anon 100 SaSu|Any1800-0600 /etc/msg.toomany
limit anon 60 Any /etc/msg.toomany
# The first two lines cause any files named README* to be printed
# if the user logs in (in this case the README* files must be in
# ~ftp) or if the user changes directory into a subdirectory
# containing a README* file.
#
# The next two lines cause the file ~ftp/etc/welcome.msg to be
# displayed on each login and ".message" files to be displayed if
# the user "cd"s into a directory containing such a file.
#
# "readme" files and "message" files are identical except "message"
# files are only displayed once per user session, whereas "readme"
# files are displayed every time the given condition is met.#
readme README* login
readme README* cwd=*
```

```
message /etc/welcome.msg login
message .message cwd=*
# Anonymous users are prompted to enter their email address as a
# password.
# "passwd-check" prints a "warn"ing message if the password doesn't
# look like a standard ("rfc822") email address. If you choose
# "enforce" rather than "warn" you will end up denying access to
# many, many people on the Internet who really don't know what
# their email address is...
#
passwd-check rfc822 warn
# Stop anonymous users from using "interesting" FTP commands.
# Directory permissions in ~ftp should also stop this behavior,
# but a little strength in depth never hurts ...
#
delete no anonymous
overwrite no anonymous
rename no anonymous
chmod no anonymous
umask no anonymous
# NOTE: Insert the correct pathnames for your ~ftp directory
# here!!! Uploads are generally not allowed in ~ftp, but are
# allowed in subdirectories of ~ftp/incoming. Files uploaded here
# will end up being owned by root, mode 4000 so that they cannot
# be overwritten by other anonymous users.
#
upload /local/ftp * no
upload /local/ftp /incoming/* yes root root 0400
# Allow anonymous users to specify filenames containing letters,
# numbers, dash ('-'), underscore ('_'), and period ('.') but not
# paths which begin with period (e.g., '../../../../../../../etc/passwd',
# which shouldn't work anyway due to chroot()) or dash (could be
# trying to play tricks with STDIN/STDOUT).
#
path-filter anonymous /etc/pathmsg ^[-A-Za-z0-9_\.]*$ ^\. ^-
# Log all commands issued by real users as an audit log. Log all
# transfers to or from this server by real or anonymous users.
#
log commands real
log transfers anonymous,real inbound,outbound
# Email address used for %E in message files
#
email admins@sysiphus.com
```

Appendix B. Welcome Message (~ftp/etc/welcome.msg)

```
This is the anonymous FTP server for <insert your company name
here>.
It is primarily for the use of the customers and employees of <your
company name>. Please do not abuse this resource. Thank you for
your cooperation.
Note that file transfers to and from this server are now logged. If
this bothers you, please log off now.
If you have any questions about this server, and especially if you
have any problems using it, please contact:

  Some Body  somebody@yourdomain.com
  Some Title +1 (666) 555-5555
  Your Company Name

If your ftp client has problems with receiving files from this
server, send a '-' as the first character of your password (e-mail
address).
```

Appendix C. Too Many Users Message (~ftp/etc/msg.toomany)

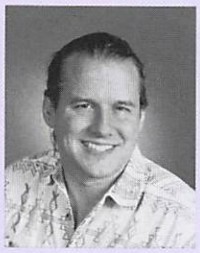
```
Sorry, there are too many users on this server at the moment.
Please try again in 15 minutes or so.

If you believe you have received this message in error, please send
email to %E.
```

Appendix D. Invalid Pathname Message (~ftp/etc/pathmsg)

```
Please choose a pathname that does not contain special characters.
```

on reliability

You and Your Users



by John Sellens

John Sellens has recently joined the Network Engineering group at UUNET Canada in Toronto after 11 years as a system administrator and project leader at the University of Waterloo.

<jsellens@uunet.ca>

This time around, let's discuss user community interaction and how it relates to reliability. Recall that we're providing services, and that users are, of course, the reason we provide those services. How can we use our user interaction to improve reliability, communicate that increased reliability is one of our goals, and help our users become part of the reliability equation? (I'll mention that I'm writing this on an airplane, and as a user of the airline's services, I very much want to know that they care about reliability, and I'm more than willing to do my part.) Let's look at the question three ways: communication to users; communication from users; and education, training, and publications.

Communication to Users

A long time ago I learned that in a number of situations it is not sufficient simply to do your job – you must also be *seen* to be doing your job. By that I mean that sometimes you must be obvious about what you are doing (and why) while you're doing it. Consider, for example, a security guard – the fact of being visible in itself can act as a deterrent and reduce the likelihood of "an incident."

Reliable system administration is another one of those tasks that are enhanced by visibility. For example, if a system is obviously being run in an organized and disciplined manner, are users more likely to act that way themselves, and thereby bring us closer to our goals? How can we be obvious about what we're doing, and why, and how can users help? A documented, predictable computing environment will be thought of as far more reliable than it would otherwise.

The standard method for successful presentations is to tell your audience what you're about to tell them, tell them, and then tell them what you've told them. There's a convenient parallel for communication to your user community:

- Tell them what you expect to do for them and what you expect from them.
- Follow through and work toward your commitments, keeping them informed of your progress (or lack thereof) and of any failures or incidents that might occur.

- Provide statistics, incident reports, and plans for improvement as you progress.

Tell Ya What I'm Gonna Do

Advance communications are often going to be the largest component of your "formal" user communication and are likely to tend toward the "static" rather than the "dynamic." You should outline (in greater or lesser detail as your situation demands) what services you will (plan to) provide – for example, centralized file service, printing, user consulting, and authentication services. The list of services will presumably have been arrived at through a process of user consultation, executive fiat, or divine inspiration (or a combination of all three), tempered by your experience and expertise and suggestions on what might be most appropriate and useful in your organization. These are your "service offerings."

Next, you should document your goals for performance and availability, both in terms of machine and network performance and availability, and in terms of guaranteed response and repair times. These are your "service-level agreements." For machines and networks, you would typically look to such metrics as percentage uptime, response times, and network latency guarantees. For example, you could state that your network file server will be unavailable less than an hour a month (99.9% uptime), that round-trip packet times between major points on your network will be less than 10ms, or that there will always be at least 5GB of available disk space (e.g., for image processing). For response and repair times, some examples might be next-business-day response for installing new personal network connections, accounts created within three hours, file restores within eight hours, first response to problem reports within two hours, etc. The important thing is to make sure that your goals and guarantees are aligned with the business needs of your organization. Figure out what services and activities are most important to your users and then determine how you can organize your resources (human and machine) to best balance those needs in delivering your services.

The last component in setting expectations is policies and practices. Reasons to establish service and usage policies include:

- conformance with, and the ability to serve, organizational goals (e.g., reserving CPU capacity for product developers working to get the next release out the door)
- making it possible to meet service-level agreements (e.g., reserving two hours on Sunday so that maintenance doesn't interrupt activities during the week)
- ensuring that users aren't interfering with services to others (e.g., no Quake during office hours)

As a service provider, you will be better off if you proactively define the policies and practices you will use to deliver your services. That is, it's better to define your own goals before some other less "reasonable" goals are imposed on you. You will typically want to consider such things as:

- standard change/maintenance windows (e.g., Saturday mornings, Tuesday nights from 11:00 pm until 3:00 am, etc.)
- emergency repair procedures, and what constitutes an "emergency"
- standard operating hours for services such as the help desk and hardware repairs
- a policy for off-hours response and a method of deciding what can wait and what has to be fixed immediately

Recall that we're providing services, and that users are, of course, the reason we provide those services.

You will need to outline the policies that your user community is expected to follow. . . . The better the understanding between the two groups, the easier it is to provide a reliable and understood service.

- a method for responding to unexpected "incidents," security-related and otherwise
- escalation procedures in case goals aren't met or problems are more substantial than they first appeared

You will also probably want to outline possible remedies or repercussions for those times when you fail to meet your stated goals and guarantees. For an internal service organization, these are more likely to involve public "humiliation," bad performance reviews, or lousy parking spots in the company lot. For an external service provider (e.g., an ISP or a computing service bureau), the result of a failure to meet the goals ar guarantees is likely to involve money.

Finally, you will need to outline the policies that your user community is expected to follow. These policies contribute to reliability by (we hope) freeing you from having to deal with malicious or unexpected acts by your users and establishing a base of understanding between users and service providers. The better the understanding between t two groups, the easier it is to provide a reliable and understood service. You will likely want to cover such areas as:

- security, password sharing, snooping, sniffing, hacking, etc.
- virus protection, and under what circumstances it is acceptable to add software to your systems and networks.
- protection of the organization's equipment (e.g., no latté with more than three suga allowed within two feet of a keyboard)
- disk-space limits, CPU hogging, and other resource-consumptive areas of contentio
- what may or may not be connected to the network, and where (inside or outside th firewall, etc.)

Change, Status, and Failure Reports

These reports should be part of ongoing, day-to-day communications with your user community. The two most important attributes of these reports are timeliness and completeness – you need to provide the information that your users want, need, or deserve at the most appropriate time.

Change reports outline a planned change to your systems or networks and its impact (or, even better, lack of impact) on your users, and summarize or confirm its implementation. These reports are important so that your users can plan their activities around expected service disruptions, and so that they can understand and prepare fo changes to software or interfaces. Reports would typically outline the expected date a time of the change, its impact, and how to obtain additional information. Internally, you should also document (in advance) how to implement that change, how to test th implementation, and, most important, how to back out of the change if necessary. Change reports should be issued far enough in advance that users have adequate prep ration and warning time, but not so far in advance that they are forgotten.

Status reports are the ongoing mechanism by which your services can be measured, both for trend analysis and to allow for evaluation against your service-level agreements. You would typically track as many of your defined metrics as possible in log files, graphs, printed reports, Web pages, summary numbers, and so forth. I recomme MRTG[1] as a terrific way to graphically track virtually anything against the calendar network traffic, uptime, users, disk space, news or mail traffic, routing-table size, nu bers of outstanding trouble tickets, and on and on. If you've got the time and you ca automate the collection and reporting (and it won't adversely affect your systems), ra

as many metrics as you can think of, even if you don't publish them all. The more information you have, the easier your troubleshooting and capacity planning will be. It's common in customer service organizations to track various service metrics (time to resolution, phone queue time, calls per hour, etc.). Many (most?) system administration organizations could benefit from more proactive status reporting.

Failure reports are, obviously, something that we would all like to keep to a minimum. They are the method by which you report system or network "incidents" to your customers, and (ideally) document your plans to ensure that such failures are reduced or eliminated in the future. They can also serve as a symbolic way for you to "take responsibility" for an outage and demonstrate your commitment to improving your service. Don't feel you need to wait until an outage is resolved to issue a failure report; your users will appreciate your openness and consideration and will also be less likely to contact the help desk about expected up times if you've already posted bulletins. The latter can be quite useful if you're with a small organization and the people responsible for answering the phones are the same people who are busily trying to fix the problem.

Track as many metrics as you can think of, even if you don't publish them all. The more information you have, the easier your troubleshooting and capacity planning will be.

Statistics, History, and Revisiting the Future

This is where analysis and prediction come into play. Using the information that you have gathered – along with projected changes in usage, information on new projects, and normal usage increases – you can generate current and historical statistics, identify trends, and predict the future. And when you've done it once, you will also be able to use it to repredict the future and compare your predictions to what actually happened. The capacity-planning uses of this information are obvious, and these reports are also a good way to demonstrate your professionalism to your users (and your management!).

Communication Initiated by Users

In order to be able to correct problems, deal with "incidents," and enable some ongoing improvement of your processes, you need to provide ways for your user community to get you the information you need (and the information that they need you to have). I've defined this communication as user-initiated in an attempt to gently remind you that communication must be two-way – you (of course) have a duty to respond (preferably in a timely and effective way).

It typically makes sense to think of two sets of user-initiated communication:

- the ongoing day-to-day problem reports, help requests, and requests for enhancement (and, if you're very good, thank you notes for a job well done)
- the periodic status reports, general reviews, direction or policy guidelines, and overall satisfaction indicators that help determine your direction and activities

The first can be expected to come from almost anyone within your organization (and sometimes from outside your organization too), while the second are more likely to come from organizational management, steering committees, user groups, and the like, as well as your own surveys and inquiries.

The most common communication method is (of course) the telephone. Most of us have probably experienced those calls out of the blue reporting some major problem, with an expectation that we will drop everything and solve it immediately. But as I review the communication process, I'll also review the alternative communication methods that you should consider and/or support.

Let's assume that you have some number of people responsible for dealing with user or

Consider the use of some form of problem-reporting form or checklist to help improve the quality of your initial data collection.

customer queries (the "help desk"), and divide the communication process into three stages: the request, tracking and resolution, and your response. (These three stages apply to both sets of user-initiated communication, but you will of course adjust your reaction and process depending on the type of communication.)

The Request

When someone needs help, or wishes to report a problem or request an enhancement, they need some "reporting method." Consider these alternatives:

Phone: a "well-known" generic problem-reporting phone number, with responsibility for answering it distributed among your help-desk staff in some reasonable fashion (queue, dedicated "on-call" person, round robin, whoever isn't busy, etc.). It is, of course, convenient if there is mnemonic value to the number (e.g., company extension 4357 – HELP), but the important things are that it exists and it's publicized. Note that some telephone systems can provide call statistics for your help-desk calls, which is handy when you're trying to prove that you need more staff.

Email: a well-known email alias, such as <help@company.com> – most of the same considerations apply here as for telephone contacts.

Newsgroup postings: a local newsgroup used to report problems or request enhancements in larger environments. (I've seen this used in a university.) This can cause the users to feel a little bit like they're sending their problem floating off like a message in a bottle, but with prompt response and tracking it can be effective. A useful side effect of this method is that it makes it possible for other users to reply, which can lessen your overall support load. This is most likely to occur in an environment like a university where large numbers of people (i.e., the students) tend to be enthusiastic generalists with a low per-hour cost and some amount of free time, or at least time available for procrastination.

Office: a consulting office or physical help desk, where users or customers can walk in and (one hopes) get helped while they wait. This is also a great place for distributing printed documentation.

Web form: the obvious method for the late 1990s, which could generate email, trouble tickets, or (potentially) even voice mail (but why would you bother?), or just about anything else. But make sure that the form is easily findable on (or from) your organization's internal Web site.

Hallway chat: Try to avoid this one, because you'll never remember all the details, you'll forget about it entirely, or something else will go wrong and get in the way of addressing the original request. (The even more problematic versions of this include the barstool chat, the running out the door "by the way, I've got a question," and the offhand comment in the washroom.) I always make it a practice to say something like "Sure, we'll get right on it, can you send some mail to `request' summarizing the situation?"

Regardless of the method used to make the request, the better and more complete the information you get with the request, the easier it will be to solve the problem; and the better you are at solving problems, the more reliable your systems and services will be. Consider the use of some form of problem-reporting form or checklist to help improve the quality of your initial data collection.

Tracking and Resolution

Once you receive a request, you must use some method to keep track of it to make sure that it doesn't get forgotten. Regardless of how simple or how sophisticated your tracking system is, it will probably involve the following six activities:

- Recording - the initial information received from the user
- Delegation - assigning the task to someone
- Tracking and note-taking - during the investigation of the problem
- Resolution - an indication that the problem or request has been fixed or addressed
- Reporting - notification to the user of resolution and ongoing efforts if still in progress
- Archiving - a work record and (buzzword alert) "knowledge base" for future reference

We often don't analyze the process in such depth, but even those little pink telephone message slips can be used as the mechanism to implement those six activities. You might, however, appreciate the added features offered by even the most rudimentary automated tracking systems.

The key benefits of a problem- and request-tracking system include:

- Reassurance for the user or customer by the assignment of a ticket number, or by query and status tools and messages if they're available. It provides an indication that you take user/customer response seriously.
- An ongoing work record, which is useful for keeping track of changes, for standard answers and fixes, and as a customer service log. It's far easier to convince management that you need more people if you have reliable statistics to back up your claim of being overworked.
- Provision of a to-do list and a mechanism to ensure that nothing gets lost. But make sure that you review the list – it won't help if you just record the request and then forget all about it.
- Enabling outstanding requests to be escalated for more focused attention (even if it's just your boss reviewing the two-month-old requests and asking you pointed questions).
- A conversation/interaction history, which makes it far easier to pass a task on to someone else and get them up to speed.
- Aid in assuring a measure of consistency in the way you respond to requests, which will lead to more effective and efficient support.
- A training tool for staff. If they can review what others have done before them, they can do better themselves.

All of these improve your reliability.

Your Response

I mentioned the need to report and reply to the requester, but it's worth repeating. The purpose of all this effort is to reply to the requester, providing advice, a fixed bug, a plan, a report, or even a statement or apology that you're unable to help (because of resource constraints, different areas of responsibility, and sometimes impossible problems). Your response should be timely, complete, and correct, and it's almost always

A modest collection of links to problem- and request-tracking systems is at <http://www.net/~jsellens/tracking/>.

One of the best ways to make yourself (or your group) more effective is to help your users and customers to help themselves whenever possible.

worthwhile to provide interim status reports if it will take a nontrivial amount of time to respond to a request. Consider the method you use to deliver your response; choose among email, paper, phone, or face to face, depending on the problem, the response, and the person who made the initial request.

Education, Training and Publications

One of the best ways to make yourself (or your group) more effective is to help your users and customers to help themselves whenever possible. A one-time investment in training materials (with a small amount of ongoing maintenance and revision) can provide a lasting positive effect on the effectiveness of all involved (service provider and service consumer). To tie this more clearly to reliability, a more effective user community will make fewer errors and will lead to a more reliable organization, and a system administrator who has fewer user requests (because of a better user education, documentation, and training program) will be better able to deal with the systems themselves, the long-term planning, and the problem avoidance that all contribute to higher reliability.

Consider some of the following mechanisms for getting the word out:

Email or Newsgroup Postings

These are probably most appropriate for notices, brief announcements, etc., and usually are not very effective as an educational tool. One exception I have seen (as mentioned above) is the use of a local newsgroup for posting and resolving problems, which often provides a good learning environment for the innocent bystanders reading the group for entertainment.

UNIX Man Pages and Other Traditional Help Systems

As more computing activity happens in a GUI workstation environment, the traditional text-based help systems and man pages are losing some relevance for end users. But these methods are still very important for system administration and other behind-the-scenes activities, and also for people working in a command-line environment. And it is of course possible to put a Web front end on traditional help text.

Web Pages

It's probably a fair bet that the vast majority of internal systems documentation is being put on the Web now. For client-server, general process documentation, help desk, desktop support, and the like, there's probably no better alternative

Local Guides

Local booklets and user guides can be very useful, as long as you have a reasonably sized topic area that doesn't need constant updating. An example might be a user and security policies document. While you would almost certainly want to make an online version available, there's still a lot of value in printed materials. I'll mention the SAGE "Short Topics in System Administration" series as a successful and effective example of local community guides (as long as you're willing to agree that SAGE is primarily a community).

One-page Guides

A number of universities and large organizations have developed very effective single-page topic guides, intended to be handed out from the help desk or consulting office. These are typically intended to be almost complete references to every (local) thing you

need to know, on topics such as setting up PPP, introduction to email, and how to print. When someone comes in to ask a question, you can provide the answer and also send them away with prewritten instructions.

Classes and Tutorials

Of course, sometimes there is no good substitute for good old-fashioned face-to-face learning, in a classroom or workshop setting. These are probably more effective for larger, more involved topics, or for areas in which it's important to get a fast start. A common example is when major new applications are put into place (such as a new purchasing system), and people have to be up and running almost immediately.

Any training or information mechanism will take time to put together and get going. Resist the urge to put it off for another day. If you can't find the time yourself, hire a contractor or a third-party firm, or buy prepackaged course materials. The time you save may be your own.

Next Time

In the next, and most likely last, "On Reliability" article, I plan to cover certain aspects of security and review how your security policies and practices affect the reliability of your systems. As always, if I've left something out, or I've forgotten your favorite topic, I'd enjoy hearing from you.

References

[1] Tobias Oetiker, "MRTG – The Multi Router Traffic Grapher," *The Twelfth Systems Administration Conference (LISA '98) Proceedings*, December 6-11, 1998, pp. 141-147.

Any training or information mechanism will take time to put together and get going. Resist the urge to put it off for another day. If you can't find the time yourself, hire a contractor or a third-party firm, or buy prepackaged course materials. The time you save may be your own.

toolman

Generating Web Pages with `sh` and `make`, Part 2

by Daniel E. Singer

Dan has been doing a mix of programming and system administration since 1983. He is currently a system administrator in the Duke University Department of Computer Science in Durham, North Carolina, USA.

<des@cs.duke.edu>

In the previous issue of *;login:*, we began to explore the topic of how the Bourne shell and `make` can be used to maintain and generate collections of Web pages. In Part 1, we covered how the shell – or practically any scripting or macro language – can satisfy the objectives of *consistency* and *simplification:* variables and functions can be defined in a master script and in individual "source" files to hold values and consolidate HTML coding. In fact, the full power of the shell can be exploited to simplify many tasks.

In this second installment, we'll discuss how `make` fits into this picture. This will not be a tutorial on writing makefiles, though relevant makefile features will be presented.

Using `make` for Automation and Mutability

We can exploit the UNIX `make` utility to great advantage, both to automate the generation of the Web pages from their source files and the master script, and to help in the production of alternative builds of the Web pages. For those unfamiliar with `make`, let me just say that it will update designated "target" files (for example, recreate a compiled binary executable) based on any changes that have been made to any source files on which they depend, and utilizing implicit or explicit production rules. See `man make` for the details.

Just as various approaches might be taken in the shell-scripting end of this process, various approaches can be taken when constructing a makefile (a specialized sort of script for `make`, usually in a file named "makefile" or "Makefile"). I've chosen to construct my makefiles in a way that makes adding a new Web page easy: I only have to add the basename of the new source file to one line in the makefile!

Here are the components that are necessary, or that at least make things a lot easier. And though there are many platform-specific capabilities and features of `make`, the ones used here should be fairly portable. Please note that I'm no `make` expert, and so there might be more elegant ways to implement some of this.

The first section of code, below, sets up macros to identify our (fictitious) Web pages. The first line is just a list of the basenames of the files, which should be the same for both those that contain the HTML source and those that contain the final HTML code. The second and third lines use a substring replacement macro feature to make lists of both the HTML source files (with a `.hsrc` suffix), and the HTML files (with a `.html` suffix). If we need to add a new Web page, we just add its basename to that first line. That's it, the exception being if a page has any special processing needs (see below).

```
NAMES = index part1 part2 part3 notes appendix
HSRCS = $(NAMES:=.hsrc)
HTMLS = $(NAMES:=.html)
```

The next section defines an implicit (default) build rule. The first two lines establish an implicit dependency of any `.html` file on an associated `.hsrc` file. The second two lines define the default command to execute to generate a `.html` file from a `.hsrc` file. This is all in cryptic *makefile-ese*, which we won't delve into here.

```
.SUFFIXES:
.SUFFIXES: .html .hsrc $(SUFFIXES)
.hsrc.html:
	./gen_html.sh $*
```

The next section defines some targets. The first target `docs` just says to make sure that all of the `.html` files are up to date. This might be the first (default) target in the makefile, and can be invoked with `make docs` or just `make`. The second one says that all of the `.html` files are dependent on the master script in addition to their default dependencies (their corresponding `.hsrc` files), and that they will all be rebuilt if it changes. The third line establishes a dependency of the file basenames on the `.html` files, and is the one that lets us specify a basename as a target; for example, `make index` instead of `make index.html`. The build rule for `index` is actually empty, but this still results in its dependencies and their dependencies (and so on) to be checked and rebuilt if necessary.

```
docs: $(HTMLS)
$(HTMLS): gen_html.sh
$(NAMES): $$@.html
```

With these definitions, rules, and dependencies in the makefile, we can now type `make` in the directory where these files reside, and any of the HTML pages that are out of date will be expeditiously updated from their respective source files, the master script, and any supplemental scripts and files. Please note that this is by no means the complete makefile. I've only included the parts here that warranted discussion. See the `man` page or other reference materials for more details on `make` and makefile fabrication. See the end of this article for URLs for a couple of sample makefiles.

Additional Dependencies

In the previous article, I mentioned the example of the alumni address databases. An additional consideration in this and similar situations is that when we update one of these databases, we also need to rebuild the `.html` files that contain the data, that is, that depend on them. It's also a good idea to update these pages if the database conversion script is modified. We can accomplish all of this by providing some additional explicit dependency lines, such as:

```
email-addrs.hsrc: email-addrs.db
      touch $@
email-addrs.html: cvt_addrs.sh
```

The first two lines say that when the database changes, the `.hsrc` file should be marked as modified. The underlying reason why we want to do this is because we want the "Last update:" line in the Web page to reflect this change, and that date happens to be based on the modification time of the `.hsrc` file. This also indirectly establishes a dependency for the `.html` file. The third line says that to the current dependencies of the `.html` file, the conversion script should be added. In this arrangement, a change to the conversion script will not alter the "Last update:" date. That's just a design decision. If we want it otherwise, we could just add `cvt_addrs.sh` to the first line, and omit the third line.

Alternative Builds

Now let's say that the versions of the Web pages in our current directory are intended as prototypes, and that the *real* pages will contain different hypertext links and will live in a different location – possibly on a totally different network and HTTP server. We can adjust our HTML source and makefile somewhat to accommodate both versions, such that we can generate either on demand.

The way that I've done this (and, again, there may be a better way) is to set up the makefile so that we can have it hard-link all of the relevant files (including itself) to a subdirectory, and then have it call itself with an argument (a macro definition) that causes an additional argument to be passed to `gen_html.sh`. This, then, tells

I've chosen to construct my makefiles in a way that makes adding a new Web page easy: I only have to add the basename of the new source file to one line in the makefile!

Got a tool that's useful, unique, way cool? Please send a description to <Toolman@usenix.org>.

gen_html.sh to use a different BASE HREF for all of the Web pages it generates, and that's about all it takes.

The additional lines in the makefile might look like this:

```
AUX_FILES = Makefile gen_html.sh cvt_addrs.sh
DB_FILES  = email-addrs.db web-addrs.db
DIST_FILES = $(AUX_FILES) $(DB_FILES) $(TMPLS)
DIST_DIR  = production-dir
...
## clear the distribution and install all links
install-dist:
     @ echo "Erasing..." ; \
          cd $(DIST_DIR) && rm -f $(DIST_FILES) $(HTMLS)
     @ echo "Linking..." ; \
          cd $(DIST_DIR) && \
          for F in $(DIST_FILES) ; do ln ../$$F ; done
## update dist html pages; assumes that links are up to date;
make-dist:
     cd $(DIST_DIR) && $(MAKE) docs DIST="-dist"
## completely wipe, install, and make the dist html pages;
all-dist: install-dist make-dist
```

And in this makefile, the rule for building the .html file will instead look like this:

```
.hsrc.html:
     ./gen_html.sh $(DIST) $*
```

When we make the distribution version with "make all-dist", make calls itself with 'DIST="-dist",' which causes gen_html.sh to be called with "-dist". All gen_html.sh has to do is catch this option, and take some appropriate actions such as setting some variables, particularly the one that sets the BASE HREF. Now all that *we* need to do is copy the newly generated set of .html files from the subdirectory to their new home!

So, maybe I've gone a bit more in the tutorial direction with makefiles than I had intended, but I feel that these examples are beneficial in building the case on the fundamental role that the make utility can play in the design and upkeep of a collection of Web pages. I hope I've convinced you.

Wrappin' Up

There're a lot of free programs and packages out there on the Web (speak of the devil) to help you to develop Web pages, not to mention the plethora of commercial packages. Some are specialized macro packages, some are WYSIWYG editors, others are server-side dynamic page generators. (I haven't used any of them; let me know if you have any recommendations.) If you care to explore, you might try searching on "HTML editors" or "HTML authoring" (or substitute "Web" for "HTML"). A good starting point for surfing might be <http://www.w3.org/Tools/>, though unfortunately this otherwise excellent page is no longer being updated.

Of course, for us do-it-yourselfers, a shell and make have a certain appeal. I'll provide some sample scripts and makefiles on the Toolman home page and FTP site; you can download them for use as starting points, if you're inclined to experiment with these ideas.

Happy scripting!

URLs:

<http://www.cs.duke.edu/~des/toolman.html>

<ftp://ftp.cs.duke.edu/pub/des/scripts/gen_html/INDEX.html>

the first ISP

It's not my fault. But sometimes I think it might be. My name is Spike. You've probably never heard of me, but in 1989 I did something that had an impact on the Internet today. I started the first ISP.

by Spike Ilacqua

Spike Ilacqua has been an "Internet professional" since 1989, first at The World in Boston, then as co-founder of Indra's Net in Boulder. He has a high school diploma and a pretty cool car.

<spike@indra.com>

Actually I didn't, Barry Shein did, I just actually made it work. In 1989 Barry started a small UNIX consulting company call Software Tool & Die. There wasn't a lot of consulting that summer, but there were a lot of calls for people who wanted to know where they could find access to email and USENET News.

You might not know what USENET News is either, but it was the reason people got online before someone thought up the Web. Lots of discussion groups with lots of people posting messages. Not in realtime, mind you. But it was like, as someone more clever than I once said, the world's biggest cocktail party.

STD (yes, I know) had really good news. Barry is, was, and always will be a USENET junkie. Word must have gotten around 'cause people who'd lost their access at school (pretty much the only way to get it back then) came nosing around looking for access. Barry had an idea: let's sell it to them.

It wasn't really a new idea. There was The Well in San Francisco, Portal might have been up and running by then, and by that time countless BBSs (which you may have never been a part of either). But nothing east of the Mississippi that we knew of.

So off we went to the BitBucket to buy six 2400bps modems (with MNP 5 and maybe Retsyn). Then came a number of sleepless nights while I wrote account-creation software, installed all the software our UNIX-hungry future customers might want, made modem cables (really) to connect up those modems to a Sun (like they made Toy Story with only much bigger and much slower and, well, only one), and drank a lot of Coke. Jim Frost wrote billing software, drank Mountain Dew, and ate Cheeze-its.

On a fine Boston (OK, Brookline) day early in November 1989, our first customer logged on and The World was born.

Those were the days. I was the sysadmin, the tech support guy, and, when Jim moved on to greener pastures (this is before there was any money in the Internet), the system programmer too. I drank a lot of Coke and ate a lot of ramen noodles.

Now, back then if you weren't an institute of higher learning or a defense contractor, you couldn't be on the "Internet." You got your email and news using UUCP, old-style. With UUCP you called other computers and exchanged mail and news with them, and they called other computers and passed it on and so on and so on. Clunky, but it worked. Someone once said, Never underestimate the bandwidth of a station wagon full of 1/4" tapes (think "minivan full of Zip disks"), and the same can be said for lots of 2400 and later 9600bps (sort of) modems.

But all things change. Sometime in 1990 the National Science Foundation decided they'd like less involvement in running the Internet (you didn't even know the NSF ran it, did you?). So they began what is popularly called, in Socialist countries, privatizing the Internet. One company to take advantage of this was UUNET, the king of the above-mentioned UUCP. They formed a new company called Alternet to provide Internet service. At this point, gentle reader, you'll be able to guess who Alternet turned to when they needed space in Boston (OK, Brookline) to install equipment.

Naturally we got connected as part of the deal.

Everything good was always somewhere else (and there wasn't even a Web yet).

But it wasn't that simple. In 1990 the NSF still ran the Internet backbone (which despite what you hear every day doesn't exist anymore) and you still had to be an institute of higher learning or a defense contractor to be connected. We could get to the parts of the Internet that were connected to Alternet and a few other networks that were privately interconnected with them (they call it peering in the modern era), but not the bulk of it. Frankly, this sucked. Everything good was always somewhere else (and there wasn't even a Web yet). Even worse, try explaining this paragraph on tech support calls all day. People always want what you ain't got.

If you've been around for a while and have read USENET, and in that unlikely event "read" something other than porn (yep, USENET has more porn that you can shake a stick at), you've probably come across Barry Shein. And if you've come across Barry you know not to argue with him or at least that you'd better have your facts straight and be damn sure of them. The man has forgotten more than I'll ever know, has a better command of the English language than Danielle Steele, drinks way way too much coffee, and somehow knows how everything comes down to the Battle of Hastings (1066 A.D.). Much of my skill as a business person comes from all those years of trying to explain to Barry why my idea was better.

Anyway, in late 1992 Barry turned his awesome powers on the NFS. And eventually they relented. The NSF allowed companies to sell dialup access to the Internet with some lame disclaimer about how those customers would probably be working with an institute of higher learning or defense contractor.

So that's how on August 13, 1992, I was running the very first official ISP. Oh, sure it was only a day or two before the other guys got their permission. Hell, by the next year the rules had been pretty much thrown out all together.

But there you have it, the first ISP.

Nine years have passed. I now run my own ISP in lovely (whiter whites, brighter colors) Boulder, Colorado. The World is still there and so is Barry, though it's now a Silicon Graphics (like they used for Jurassic Park). When they write the history books of the Internet, my name won't come up. Maybe The World will, maybe it won't. If it hadn't been me it would have been someone. But *I* was there and *I* did the deed and somehow, just maybe, as I filter the spam out of my inbox and see the Web address on the side of trucks and cans of soda, it may be partially my fault.

java performance

Memory Fragmentation

by Glen McCluskey

Glen McCluskey is a consultant with 15 years of experience and has focused on programming languages since 1988. He specializes in Java and C++ performance, testing, and technical documentation areas.

<glenm@glenmccl.com>

Suppose that you are developing an application, one that will make use of very large arrays of objects. If you've studied the area of memory allocation at all, an issue that may be familiar is fragmentation: Adequate memory is available to an application, but the memory is broken into small chunks, none of them big enough to satisfy a given memory request. For example, the application may request 1000K of contiguous memory to represent some data structure, and the request will fail because only four chunks of 250K each are available.

This problem has an obvious solution, which is to represent the data structure in pieces and map indices into the structure into the appropriate piece. Such a technique will work with any programming language, but languages like C++ and Java make it easier, because the details of representation and mapping can be encapsulated within a class.

To see how this works, consider a Java class to represent very large or sparse arrays of objects:

```
import java.util.*;

public class SparseArrayList extends AbstractList {
// default page size
private static final int DEFAULT_PAGE_SIZE = 1024;

// actual page size that is set
private int pagesz;

// pages
private Object pages[][] = new Object[0][];

// default constructor
public SparseArrayList()
{
   this(DEFAULT_PAGE_SIZE);
}

// constructor specifying page size
public SparseArrayList(int sz)
{
   if (sz < 1)
      throw new IllegalArgumentException();
   pagesz = sz;
}

// number of slots currently allocated
public int size()
{
   return pages.length * pagesz;
}

// set an array slot to a given value and return the old value
public Object set(int index, Object val)
{
   if (index < 0)
      throw new IllegalArgumentException();

   int p = index / pagesz;

   // if page array not big enough, expand

   if (p >= pages.length) {
      Object newpages[][] = new Object[p + 1][];
      System.arraycopy(pages, 0, newpages, 0, pages.length);
      pages = newpages;
   }
```

```
    // need to allocate a new page?
    if (pages[p] == null)
      pages[p] = new Object[pagesz];
    Object old = pages[p][index % pagesz];
    pages[p][index % pagesz] = val;
    return old;
  }
  // get the value of a given array slot, null if none
  public Object get(int index)
  {
    if (index < 0)
      throw new IllegalArgumentException();
    int p = index / pagesz;
    if (p >= pages.length || pages[p] == null)
      return null;
    return pages[p][index % pagesz];
  }
}
```

Java 2 contains a set of classes known as a "collections framework," and the `SparseArrayList` class is based on this framework. `Collection` and `List` are top-level interfaces (an interface is a specification of a set of methods that an implementing class must declare and implement) in the framework, and the abstract classes `AbstractCollection` and `AbstractList` partially implement the functionality of these interfaces. `SparseArrayList` need only define constructors and the `size()`, `get()`, and `set()` methods in order to fully implement the interfaces.

`SparseArrayList` represents an array as a series of pages, each containing a default of 1024 elements. An array index is split apart to find the page and offset within the page. A page is allocated only if needed. An array of 1 million elements will use 977 pages (1 million / 1024), so the array will be represented as a page array 977 long, together with a series of object arrays each 1024 long. Note that specifying very small page sizes may result in frequent reallocation and copying of the pages array, hurting performance. One solution to this problem would be to define a constructor that specifies the maximum size of the array in advance, and another solution would be to grow the pages array by more than one page slot at a time.

A simple program that exercises this class is:

```
import java.util.*;
public class testarray1 {
  public static void main(String args[])
  {
    Random rn = new Random();
    List sal = new SparseArrayList();
    for (int i = 1; i <= 1000000; i++) {
      // generate a random number
      int r = rn.nextInt(1000000);
      // set its array slot and then
      // retrieve and compare
      sal.set(r, new Integer(r));
      if (((Integer)sal.get(r)).intValue() != r)
        System.err.println(r);
    }
  }
}
```

There are obvious advantages to using a class to encapsulate a data structure. But what about the costs of doing so? Accessing an array through a method is slower than using primitive machine operations. One answer to the question is to come up with a simple benchmark:

```
import java.util.*;
public class testarray2 {
  public static void main(String args[])
  {
    List sal = new SparseArrayList();
    Object obj = new Object();
    for (int i = 1; i <= 1000000; i++) {
      sal.set(i, obj);
      obj = sal.get(i);
    }
  }
}
```

This program sets and then gets 1 million elements from a `SparseArrayList` structure. It requires around 1.35 seconds to run on a 300MHz Pentium, or 1.35 microseconds per iteration. For 8 million elements, the time increases to 1.7 microseconds per iteration, reflecting the extra time to reallocate the page table.

One final point about `SparseArrayList`. Because it is designed to operate within the collections framework described above, an object of type `SparseArrayList` can be specified as a method argument wherever a `Collection` or `List` is called for, and standard operations such as iterators are automatically available. For example, the following program creates a `SparseArrayList` and then sets the values of the first few elements. Then the element values are dumped out via an iterator.

```
import java.util.*;
public class testarray3 {
  public static void main(String args[])
  {
    List sal = new SparseArrayList(10);
    sal.set(0, new Integer(0));
    sal.set(1, new Integer(1));
    sal.set(2, new Integer(4));
    sal.set(3, new Integer(9));
    sal.set(4, new Integer(16));
    sal.set(5, new Integer(25));
    Iterator iter = sal.iterator();
    while (iter.hasNext())
      System.out.println(iter.next());
  }
}
```

The output is:

```
0
1
4
9
16
25
null
null
null
null
```

The iterator mechanism uses `size()` and `get()`, and thus the values for the whole page are reported.

There are obvious advantages to using a class to encapsulate a data structure. But what about the costs of doing so?

intrusion-detection systems

Guaranteeing the Safety of a Network Beyond Using a Firewall

by Dario Forte

Dario Forte is an Italian system administrator. He is CCSE and CCSA CheckPoint Certified and is a USENIX-SAGE-CSI individual member. In Italy he moderated the independent forum of Windows NT Security and CheckPoint Firewall-1.

<dario.forte@computer.org>

The increase in the number of links with the Internet has generated a proportional increase in the number of attacks brought against them. Implementing systems capable of detecting these attacks (possibly without false alarms, which in some cases is an impossible dream), and also giving the system administrator the opportunity to intervene immediately, is a specific area of networking research: intrusion detection. This article discusses how and why intrusion-detection systems (IDSs) should be used to record and report possible violations of information systems linked to the Internet.

IDSs can be considered an extension of network analyzers. Network analyzers are devices that monitor and analyze the network traffic in realtime, for the purpose of identifying any arbitrary violations of general policy established by the network manager. The differences among them concern the methods used for analyzing and reporting, and the type of traffic that can be identified. The task of an IDS is to record and report any violations of information systems (even in the form of attempts). They are not permanent devices. In other words, they do not constitute a replacement for a firewall, a proxy, or similar devices. IDSs are installed in addition to firewalls and carry out a check at an internal level, on the customer side of the network.

Generally speaking, IDSs are grouped into two families: the first performs mainly auditing functions, carrying out continuous calls to the hosts linked to the system. The second works independently, observing the traffic on the networks directly (and passively), carrying out "usual" packet filtering. Some commercial products combine the two methods.

Many IDSs feature automatic network-attack recognition and response-system type. The IDSs are installed and made to "run" throughout the network or on parts of it where there is a need to preserve critical information.

Monitoring of data traffic generally takes place on the TCP/IP stream passing through Ethernet-based infrastructures. (Some manufacturers are working on supports for different layouts.) Thus, the IDS combines the functions of a simple network analyzer with that of recognizing attacks. This is generally achieved by means of a signature check, rather like programs that detect viruses. The system contains a database of attacks, which functions as a basis of comparison with the traffic analyzed. When an attack is recognized, the IDS can stop the stream of data immediately, thereby preventing the attack from causing damage to the network.

An IDS that does its job properly must also enable recording of the date, source, target, and type of the attack, and of the activity undertaken. As for the source of the attack, the IDS must implement antispoofing functions. This turns out to be very important, primarily for preventing some DOS (denial-of-service) attacks.

The basic principle of operation of IDSs is prompt intervention. In practice, once the attempt to intrude has been discovered, the system must warn the person in charge of corporate security, who must be able to implement the necessary countermeasures, even from a remote location. It is possible for the system itself to give a warning of the

incoming attack, for example by means of an SMS message (texts on GSM) or a warning sent to a pager for the security administrator.

As to required computing capacity (ascertainable in terms of the slowdown it causes in the performance of the whole network), an IDS must be able to facilitate maximum scanning speed and therefore present minimum requirements in terms of latency. Although this is guaranteed by most manufacturers, discussions among administrators on this issue are focused mainly on this aspect.

From the point of view of development philosophy, the most widespread analysis model used by the IDSs currently being circulated is called CIDF (Common Intrusion Detection Framework). This model consists of splitting up the IDS into a series of components, called boxes, which range from the management of suspicious occurrences, to the countermeasures adopted, to safe storage of the logs generated. These boxes, which are generally preceded by a letter indicating their function, are of fundamental importance also for legal purposes; regulations concerning computer crimes may require the log files as proof of illegal access.

The task of an IDS is to record and report any violations of information systems (even in the form of attempts). They are not permanent devices.

Why Get an IDS?

While most attacks come from outside, badly administered and controlled internal resources can be a source of large-scale problems. That is the main reason why, at times, installing a firewall is not sufficient to limit the damages. In addition, a firewall can crash or, worse still, it might have been improperly configured. In that case, it is important to have a rearguard product able to interact with the perimeter of the system.

Computing Demands and Band Requirements

It must be possible to control the central unit and management from dedicated machines. For this purpose, for example, RealSecure Engine requires a minimum 200MHz Pentium, 32MB of RAM, at least 199MB of disk space for the log files and databases, 10MB for the software, and an Ethernet NIC operating in promiscuous mode. A remote management console, on the other hand, requires, for an IDS working on Intel systems, a 200MHz Pentium, 32MB of RAM, and 100 MB of disk space for each scanning engine it manages.

Arguments about the Effectiveness of IDSs

To what extent can an IDS influence the performance of a network? How effective can it really be? Management wonders most about such issues. Two well-known American security experts, Thomas T. Pracek and Timothy N. Newsham, have recently triggered a discussion about the true effectiveness of IDSs. According to these two experts, these types of device have many "weak points." However, most of these Achilles' heels reside in the analysis models.

In considering IDSs based on the signature-analysis method, also known as misuse detection, Pracek and Newsham have raised the objection that often the policies implemented are too stiff and are linked to mere signature matching; that is, in practice they are based excessively on passive monitoring. Errors made during configuration can lead to a burst of false alarms, particularly when specific services are used. To this end, the two experts, who state that they are more in favor of the active-proxy monitoring type of setup, feel that maximum granularity of the IDSs based on this method is required. As they have pointed out, highlighting the possible leaks in the analysis and operational model of an IDS does not label these devices as irreparably unsafe – not at all. Instead, it points to a series of improvements to make to products in order to make these systems as safe as possible.

Some analysts compare IDSs with the first virus-detection programs.

Conclusions

The analysts feel that the current status of IDS technology can be compared to that of firewalls a few years ago. Some of them compare IDSs with the first virus-detection programs, with all the problems they used to have. Perhaps, they stress, the solutions available on the market are still "immature." It should be considered, however, that products such as those surveyed in the sidebar will have something authoritative to say within a short time, or might even become reference points. Apart from anything else, I feel that the optimization process of these systems will be accelerated to an extent that is directly proportional to the commitment the international academic community will put into studying the problems involved.

A Survey of the Best-known IDS Products

CyberCop

CyberCop, originally produced by Network General, is an IDS recently released by Network Associates <www.nai.com>, an organization that is constantly at work acquiring technologies in the field of security. After their acquisition of Network General's product, Nai made some changes to CyberCop. (It must be remembered that the auditing/scanning software now called CyberCop Scanner – also known as Ballista – is different from the totally software-based product we are discussing here.)

Installing CyberCop does not require the network to be reconfigured or plug-ins to be added. Like other IDSs, CyberCop builds a layer of additional software which works by monitoring the ports and services enabled by the firewall.

The first version of CyberCop, announced in 1997, consists of two elements, the management server and the sensors. The latter are positioned at strategic points on the network and communicate any suspicious events to the management server. These events are classed according to a set of 170 different attacks.

If an attempt is made to access the network, the product, currently called CyberCop Server, informs the security administrator in real time, providing a detailed report of the event. The designers feel that within a few minutes CyberCop can give the input the security manager needs to take the necessary steps to resolve the problem. Management of the configuration of CyberCop, as well as the receiving and transmitting of the intrusion detection reports, can take place from a remote location using an encrypted link, which is activated only after recognition of the parties.

Of course, all the traffic monitored is stored in log files which can be consulted at any time by the security manager, both in order to trace the attacks and in order to take subsequent legal action. Configuration and positioning of the sensors are simplified by a preconfigured installation set, which makes operation easier and enables leaks to be limited.

Bro

Bro is a realtime IDS devised and developed by Vern Paxson and other experts at the Network Research Group of the Lawrence Berkeley National Laboratory. The source code of Bro is freely available, and the principle on which it is based is decidedly in an academic mold. With its spartan interface, indicating that greater attention has been paid to substance than to appearance, Bro bases its operational capacity on its scanning speed, realtime notification of violations, and a clear separation between the engine, the policy implemented, and the extensibility options.

Bro is partitioned into two components: the "event engine," which translates the traffic intercepted at kernel level into high-level events, and the "policy script interpreter," which defines the policy implemented, always by means of specific instructions written in a proprietary language. In this way, administrators can use the granularity of this IDS to adapt the system to their own requirements. The services monitored on a priority basis by Bro are Finger, FTP, and Telnet. In addition, the Portmapper function of this solution makes it possible to check the activity of the single ports as well.

So far we could say that there is nothing new. All network analyzers (or net sniffers, if you prefer), and therefore all IDSs (which can be considered extensions of them) are normally equipped with these features. The designers of Bro, however, state that during the analysis period they studied in depth the typology of both standard attacks and those that can be brought to bear on the screen in the narrowest sense, and that they were able to identify and describe attacks not referred to in the literature. Again, during the prebuilding phase the designers acquired substantial experience with systems based on offline analysis of tcpdump attempts. All this has given rise to a melting pot of reference information for subsequent implementation of the modules of this IDS.

One of the main objectives of Bro is to ensure traffic speed. In order to do this, Bro monitors DMZ links. These are usually FDDI, so that the monitor must be able to inspect the traffic, which is very bulky in itself, at speeds in excess of 100 Mbps.

Bro's separation of the engine from the rest of the modules, including the script policy interpreter, is essential to streamline the monitoring operations as much as possible (which means no degradation of network performance) and to distinguish the data on the basis of the services to which they belong. All this has been implemented in order to give Bro maximum flexibility. (Flexibility is more or less the reason why the manufacturers of virus-detection programs are revolutionizing the way they are developed. Attacks are becoming increasingly numerous and diversified, and they depend more than ever before on flaws in individual operating systems and their layouts, so they are increasingly well-targeted and unforeseeable. In this context, modularity and extensibility are strategic and can only be achieved if the architecture used is as open as possible.)

More information about Bro may be found at <http://nrg.ee.lbl.gov/nrg/papers.html>.

ISS RealSecure

ISS RealSecure for Windows NT by Internet Security Systems <www.iss.net> is one of the best known and best-selling IDSs on the market. (Indeed, ISS and CheckPoint have joined in partnership to bundle Firewall-1 and RealSecure together.) The basic operating principle is common to the other IDSs: The traffic passing through is monitored, and the activities are compared with the pattern with which it is outfitted. In the event that they match up, an alert is activated and possible automatic countermeasures are implemented.

Suspicious activities, documented with information concerning the chronology of the attack, its source, and destination – plus other data to be selected – can be managed extremely dynamically. Monitoring of the traffic consists above all of packet filtering. It is possible to configure RealSecure to check traffic in all its meanings: TCP, UDP, ICMP, source and destination ports, etc. It is also possible to check the traffic on the basis of the services used because the pattern of the attacks follows this schematic distribution.

The designers of ISS used this philosophy: Starting from the assumption that most attacks come from inside, the administrator needs a product able to check all the traffic (not only the traffic permitted by the perimeter security system). In addition, a check of the activity permitted by the firewall is also indispensable, since even an authorized user can "penetrate" a system.

The security policy set up by RealSecure therefore has the objective of checking and identifying beforehand:

- who can access the system and who cannot
- which protocols and/or services are permitted
- which new hosts are added to the network and what rights they have to "dialogue" with the rest of the infrastructure.

Starting from these assumptions, a series of features in Real Secure are aimed at making the work of the administrator as easy as possible and the system as flexible as possible.

NID

NID 2.x is an intrusion-detection suite available freely on the Web <http://ciac.llnl.gov/ctsc/nid/> for various operating systems, including LINUX, but its use is limited, for the moment, to government organizations.

NID works in a manner similar to Bro. It can monitor speeds and layouts, including FDDI and, of course, all IP traffic. NID has these features:

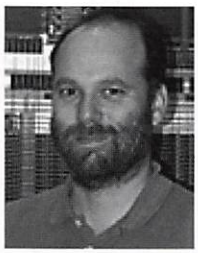
- The software is installed on a dedicated machine.
- A security domain is formed from the management console. In turn, this includes a series of hosts at the discretion of the operator.
- NID starts to audit the network traffic using three fundamental methods.
- Attack-signature recognition
- Vulnerability risk model, i.e., general safety parameters to be observed
- Anomaly detection, i.e., recognition of abnormal behavior inside the network and immediate notification of the system administrator

In NID, too, the analysis model and its operational expression are of the mainly "passive" type; traffic is audited and a consequent comparison with the attack pattern at disposal is made. If a match is found, an alarm is sent to the security administrator.

The software permits sessions of specific UNIX tasks, such as cron, to be run.

speedier squid:

A Case Study of an Internet Server Performance Problem



by Jeffrey Mogul

Jeffrey Mogul, of Compaq's Western Research Lab, currently works on Internet and operating system performance issues. He was program chair for the Winter 1994 USENIX Conference,and is a co-author of the HTTP/1.1 specification.

<mogul@pa.dec.com>

I couldn't have done this work without help from Kathy Richardson and Carlos Maltzahn, who set up the proxy software; Gaurav Banga, whose work on `select()` performance got me started on this problem; the DCPI team, who produced a wonderful tool; and Duane Wessels and the Squid team, for providing the Squid software. Jeff Dean and Gaurav helped with proofreading.

It's not always easy to discover what limits the performance of an Internet server application. Is the network overloaded? Is a Web page too complicated? Is the operating system inefficient? Is the application doing something the wrong way?

People make guesses about why a server runs slowly, and their intuition is often very wrong. Internet servers, such as Web servers and proxies, are a particularly tricky domain for intuition, because their performance depends on so many different aspects of computer system design. We may not even have a good feel for how fast a particular server ought to be, let alone why it runs slower than we would like.

This article is a case study of one such performance problem and what I did to solve it. Of course, I'll discuss the solution, but my main goal is to show how one can combine an advanced performance-measurement tool and some understanding of computer performance to rapidly discover and solve a performance bottleneck.

What Is Squid?

Squid[7] is a Web (HTTP) proxy server. It was based on software from the earlier Harvest research project[4] and is being developed and maintained by a community of Internet users, under the supervision of the NSF-funded National Laboratory for Applied Network Research. The Squid source code is freely available and runs on a wide variety of UNIX-like systems, making it a convenient platform for research; as a result, Squid is one of the most widely used proxy-server applications, even in nonresearch environments.

The primary goal of the Squid project is to provide a Web caching service. Squid servers are typically used as proxy caches, storing previously retrieved Web objects for later reuse. However, Squid can also be used as a noncaching proxy, typically as part of a security "firewall" between an organization and the Internet. For example, the Palo Alto firewall between the Internet and Digital Equipment Corporation's internal networks has used Squid as both a caching proxy and a noncaching proxy. (Although Digital has merged with Compaq, most or all of the users of this proxy are in the ".dec.com" domain.) My experiments were performed at this firewall.

When Squid is used as a caching proxy, its performance is greatly complicated by its use of disk I/O for storing, finding, and retrieving cache entries. In some contexts, such as the Digital Palo Alto firewall, where external network connectivity is reasonably fast, the disk I/O overhead of caching turns out to be much higher than the latency improvement from cache hits (which could be the subject of a future case study!). Thus, we typically run the Squid server in its noncaching mode.

A noncaching proxy makes little use of the filesystem, but it still faces performance bottlenecks. As the HTTP request rate increases, so does CPU utilization. Since our proxy often handles 2.5 million requests on a busy day, with peak rates much higher than the average, it can't afford to spend much CPU time per request. More important, we want it to scale well: The CPU time per request should not increase significantly as the request rate increases. For example, if the cost per request is proportional to the request rate, then the system's cost will increase in proportion to the square of the request rate (called $O(N^2)$ behavior).

The use of Squid as a noncaching proxy stresses many aspects of the system, including those that also affect Squid caching-proxy performance. Thus, although this article

concentrates on noncaching performance, the results are still applicable to a caching Squid system.

For these tests, I used Squid version 1.1.20. Later versions have been released, but this is relatively close to the current "stable" version of the software.

Lab Tests versus Live Tests

Many Internet performance tests are conducted in laboratory environments. For example, the SPECweb96 benchmark, used for measuring Web server performance, assumes that the clients are connected via a LAN. This assumption may be necessary for a feasible benchmark test (because it is expensive and difficult to perform repeatable measurements at high request rates in the real Internet), but it can seriously skew the results.

In lab tests of Web servers, connections typically last for a few tens of milliseconds. However, HTTP connections in the real Internet often take hundreds of milliseconds, or even several seconds, because of much longer round-trip times, higher packet loss rates, and lower link bandwidths.

Consider a server handling 100 requests/sec. If the mean request duration is, say, 25 msec., then on average the server will be handling 100 x 0.025 = 2.5 connections at once. But if the mean request duration is 2 sec. (not atypical for the real Internet), then the same request rate means that the server will be handling, on average, 200 connections at once. If the server application uses algorithms that scale badly with the number of connections, or if it uses operating system features that scale badly, a request load that is easily handled in the lab may overwhelm the system in live use.

The measurements in this article were made on our live proxy system, in order to reflect real-world performance issues accurately. This makes it harder to obtain repeatable results and so requires some attention to experimental procedures. For example, trials must last long enough to avoid short-term fluctuations in load. But since the request load varies by a factor of ten between "prime time" (usually late morning on weekdays) and the middle of the night, it's also important to decide whether one wants to measure performance over a wide range of offered loads, or during a period of relatively constant load. I used one-hour prime-time trials when I wanted to measure peak-load performance, and full-day trials when I wanted to measure performance over a wide range of loads.

Tools in action

Performance measurements are only as good as the tools one uses to obtain them. I'll briefly explain the tools that I used.

Measuring Load and CPU Utilization

Since my primary concern was how much CPU time the Squid system uses and how this scales with request rate, I needed to measure CPU utilization over short periods of time and then correlate these measurements with the request rate.

Most Web servers, including Squid, can record per-request information in a log. We already had software to analyze the Squid request logs (offline, so as not to add load to the server) and compute request rates over various intervals. I chose a 15-minute interval, which yields 96 samples over a 24-hour period.

Our servers run shell scripts every 15 minutes to log various system statistics (including output from netstat, ps, vmstat, etc.). Although vmstat does display CPU utilization (by fraction spent in user mode, kernel mode, and idle time), in order to get 15-minute

My main goal is to show how one can combine an advanced performance-measurement tool and some understanding of computer performance to rapidly discover and solve a performance bottleneck.

The CPU-utilization measurements show how much CPU time is being spent at various request rates, but they don't show where it is being spent. . . . To localize bottlenecks more precisely, one needs a procedure profile.

totals for these fractions it proved simpler to write a short program that provides the total number of clock ticks spent in each mode since the system was booted. A postprocessing script computes the differences between subsequent samples, avoiding any errors introduced by clock skew in the sampling process.

With measurements for both the request load and the CPU utilizations over identical 15-minute sampling intervals, I could then plot the relationship between request rate and CPU utilization. For example, see Figure 4. I could also run linear regressions to quantify the relationship. For example, see Table 2.

DCPI for Procedure Profiling

The CPU-utilization measurements show how much CPU time is being spent at various request rates, but they don't show where it is being spent (aside from the coarse breakdown of kernel-mode time versus user-mode time). To localize bottlenecks more precisely, one needs a procedure profile. Profiling software has been around for many years, going back to early versions of UNIX[6]. However, most profiling tools have several drawbacks:

- They cover only one application at a time (and don't include kernel functions).
- They count instructions executed, not cycles spent during execution.
- They require recompilation or post-processing of application binaries.
- They may add significant execution overhead.

A system called DCPI[1, 5] (for "Digital Continuous Profiling Infrastructure") has none of these drawbacks. DCPI, which is available for Compaq's Digital UNIX and Windows NT on Alpha processors, consists of a dynamically loaded kernel device driver and a set of other programs. It is capable of profiling the entire system, including the kernel, shared libraries, and all applications, while adding an overhead of only 1% to 3%. DCPI counts cycles, not instructions, so it accounts for the cost of delays such as cache misses and TLB misses.

This makes DCPI an effective tool for profiling an Internet server, since it quickly and easily reveals the functions that account for CPU consumption, no matter where they lie. Table 1 is a DCPI profile of the unmodified Squid 1.1.20 software running on a modified version of Digital UNIX V4.0B, on a 500MHz AlphaStation 500 system (21164A processor, SPECInt95 = 15.0) with 512MB of RAM. The table includes all procedures that account for at least 0.5% of the non-idle CPU time. The kernel modifications, described in another paper[3], improved the performance of the kernel's `select()` system call, although Table 1 shows that the various kernel procedures that implement `select()` still account for about 25% of the non-idle CPU time.

However, the single most expensive procedure, in this profile, is the `comm_select()` procedure in the Squid program. In fact, no other nonlibrary procedure in Squid accounts for more than 0.69% of the nonidle CPU time, and all of the other Squid procedures together account for less than 5% of the non-idle CPU time. This suggests that `comm_select()` is the best target for further analysis.

DCPI for detailed analysis

Suppose we want to compute the number of CPU cycles required to execute a particular piece of code. We could start by counting instructions. However, most high-performance CPUs are able to issue multiple instructions in a single cycle, although resource limits and operation latencies usually prevent the CPU from filling all issue slots on every instruction.

More important, modern CPUs experience "dynamic stalls," where they must stop issuing instructions for a short period. For example, if the CPU encounters a conditional branch instruction and has incorrectly guessed whether the branch will be taken (a "branch mispredict"), its instruction pipeline must be refilled before it can continue at full speed.

The lengthiest dynamic stalls usually occur when the memory system is unable to deliver data or instructions fast enough. While caches can help, most real applications don't fit in on-chip caches. Main memory latencies are often measured in hundreds of nanoseconds, while fast CPUs have cycle times of 2 nanoseconds or less, so one cache miss can cost more than several hundred instructions. This is especially problematic for operating-system code and similar programs, such as Squid, that to do relatively little arithmetic and that manage complex data structures.

DCPI's procedure-level profile can include the number of data-cache (D-cache) misses encountered by a procedure (not shown in Table 1). DCPI found that `comm_select()` encountered far more D-cache misses than any other procedure. Perhaps something in `comm_select()` is causing lots of dynamic stalls?

DCPI includes a tool called `dcpicalc` that uses the information in a profile, together with an accurate model of the CPU, to deduce which instructions in a procedure are stalling, and why. The full `dcpicalc` output for `comm_select()` is too lengthy to include, so I will show some short excerpts.

Total CPU %	Non-idle CPU %	Procedure	Mode
65.80%		all idle time	kernel
34.20%	100.0%	all non-idle time	
10.30%	30.12%	comm_select	user (Squid)
8.42%	24.63%	all select procedures	kernel
1.23%	3.60%	all TCP procedures	kernel
1.20%	3.52%	malloc-related #1	user(libc)
1.12%	3.28%	malloc-related #2	user (libc)
0.70%	2.05%	in_pcblookup	kernel
0.60%	1.76%	malloc-related #3	user (libc)
0.45%	1.31%	malloc-related #5	user (libc)
0.32%	0.95%	bcopy	kernel
0.29%	0.84%	read_io_port	kernel
0.28%	0.81%	syscall	kernel
0.27%	0.80%	reg_save	kernel
0.27%	0.80%	memset	user (libc)
0.27%	0.79%	malloc-related #4	user (libc)
0.24%	0.71%	hardclock	kernel
0.24%	0.70%	_doprnt	user (libc)
0.23%	0.69%	memCopy	user (Squid)
0.20%	0.60%	_XentInt	kernel
0.18%	0.53%	strcspn	user (libc)
Profile on 1998-07-16 from 11:00 to 12:00 PDT			
mean load = 17 requests/sec; peak load ca. 73 requests/sec			

Table 1. DCPI profile of unmodified Squid

A common measure of CPU efficiency for a particular pieces of code is the average "cycles per instruction" (CPI). For example, a dual-issue machine with no stalls can, at best, execute at 0.5 CPI. Dcpicalc found, by analyzing the `comm_select()` profile against a model of the Alpha 21164A CPU, that the best-case execution could have averaged 0.71 CPI, but the actual execution averaged 2.69 CPI.

`Dcpicalc` then analyzed each profiled instruction to deduce how often it stalled, and why. This analysis is complex and based somewhat on heuristics, but generally gives plausible results. It found that 71.4% of the cycles spent in `comm_select()` went to D-cache misses, and only 3.7% went to other dynamic stalls. What is causing those D-cache misses?

Figure 1 shows excerpts from `dcpicalc`'s annotation of the worst basic block in the `comm_select()` profile. (A basic block is a set of instructions always executed as a group.) Dcpicalc starts by saying that this block could have executed at 0.67 CPI in the absence of any dynamic stalls, but actually executed at 8 CPI. It then gives, for each instruction:

- the instruction address
- the source-code line number
- the opcode and operands
- the best-case number of cycles (which might be zero, for instructions eligible for multiple-issuing)
- The estimated actual number of cycles (which also might be zero).

```
*** Best-case 4/6 = 0.67CPI, Actual 48/6 = 8.00CPI
*** (8% execution without dynamic stalls)
015798 1011: ldl    t5, 156(t2)        1   1.0cy
01579c 1013: and    t0, 0x1f, a2       0   1.0cy
0157a0 1013: bis    zero, 0x1, a3      1   1.0cy
0157a4 1013: sll    a3, a2, a2         1   1.0cy
0157a8 1011: cmple  t5, t6, t5         1   43.5cy
0157ac 1011: beq    t5, 0x1200157c0    0   0
```

Figure 1. DCPI analysis of the worst basic block in `comm_select()`.

Note that the `cmple` instruction takes, on average, 43.5 cycles. `Dcpicalc`, in cryptic annotations not shown in Figure 1, also tells us that the most likely cause of this delay is a D-cache miss, that the instruction is probably waiting for data loaded from memory by the `ldl` instruction, and that this instruction was generated from source line 1011.

```
 1008   for (i = 0; i < maxfd; i++) {
 1009       /* Check each open socket for a handler. */
>1010       if (fd_table[i].read_handler) {
>1011         if (fd_table[i].stall_until <= squid_curtime) {
 1012           nfds++;
 1013           FD_SET(i, &readfds);
 1014         }
 1015       }
>1016       if (fd_table[i].write_handler) {
 1017         nfds++;
 1018         FD_SET(i, &writefds);
 1019       }
 1020   }
```

Figure 2: Source-code excerpt from `comm_select()`

`Dcpicalc` shows similarly lengthy delays at source lines 1010 and 1016. Now we know the location of the problem (or at least *a* problem): lots of D-cache misses at source lines 1010, 1011, and 1016 in `comm_select()`. Figure 2 shows that neighborhood of the source code.

From this code excerpt, it's pretty clear what is happening. `Comm_select()` is building up two bitmaps for use as input to the `select()` system call, by iterating over an array of structures, one entry per file descriptor. The loop looks at three fields in each array entry, and is getting a D-cache miss on many (or perhaps all) of these references. `fd_table` is declared so that the three fields used in one iteration of this loop are not adjacent, and might even lie on separate cache lines.

Because the number of active file descriptors is roughly proportional to event rate, and the cost of executing this loop is proportional to the number of file descriptors, the cost per event is proportional to the event rate. This causes $O(N^2)$ behavior, which prevents efficient scaling.

Statistics from our proxy server show that the `maxfd` variable, the upper limit of this loop, can reach values of 2380 or more, so each complete execution of this loop incurs as many as 2380 x 3 = 7140 D-cache misses. No wonder `comm_select()` takes so much time.

Fixing the Problems

Since I'm basically lazy, my first attempt to eliminate these D-cache misses was to simply change the order of the fields in the declaration of `fd_table`, so that the three fields referenced on each loop iteration were adjacent, and then recompile Squid. This had the advantage that it shouldn't require any debugging, and because the Alpha 21164A has 64-byte cache lines, it should result in just one cache miss per iteration.

That modification reduced the cost of `comm_select()` to 14% of the nonidle CPU cycles (from 30%). The average CPI for this procedure decreased from 2.69 to 2.43, mostly by reducing the number of cache misses on source line 1016. The discrepancy between the small change in CPI and the somewhat larger improvement in total time may reflect variation in Squid's request rate, which was significantly higher during the second trial.

Making the three interesting fields adjacent in each array element does improve the D-cache miss rate, but because the array elements contain many other fields that are not accessed during this loop, the D-cache is still filled mostly with useless information. I tried another small modification, creating a separate array to hold just these three fields. I reasoned that this would make full use of the D-cache during the loop. The result was a dramatic decrease in the CPI of `comm_select()`, to 0.75, but only a small decrease in the non-idle CPU cycle utilization of `comm_select()`, to 11%.

This was frustrating. I had been able to eliminate most of the delays directly associated with D-cache misses, but the procedure was still taking more of the CPU than anything except the kernel's `select()` implementation. I finally admitted that I would have to try an algorithmic change, at the risk of more extensive debugging.

The goal of the loop in question is to create two bitmaps for use by `select()`, based on the state of the entries in the `fd_table`. I changed the code to keep the bitmaps as long-lived data structures, updating them whenever the fields in `fd_table` are updated. Because `select()` destroys its input bitmaps, one actually has to make a copy of each bitmap before calling `select()`. Even accounting for the cost of this copy, the size of the data referenced before each call to `select()` decreases by a factor of about 40 (because the Alpha uses 64-bit pointers; on a 32-bit CPU, the decrease is by a factor of 24).

One complication is that the loop compares the `stall_until` field against the current time, so the `readfds` bitmap cannot simply be updated just when the `read_handler` field changes. However, these timestamps have a granularity of one second, and the `comm_select()` function already iterates over the entire `fd_table` once per second (to check for other timeouts), so it adds almost no overhead to also do the `stall_until` comparisons once per second.

With this change made (and debugged), the mean CPI for `comm_select()` declined again, to 0.61, and its fraction of the nonidle CPU cycles declined to just 3.7%.

Further work with `dcpicalc` showed that most of the remaining cycles in `comm_select()` were being spent on the second line of the code excerpted (in a heavily abstracted form) in Figure 3. This loop is going through the bitmaps returned by `select()`, looking for any "set" bits, and calling the corresponding handlers. The `FD_ISSET()` macro, which extracts a single bit from a large bitmap, executes numerous instructions, and so this loop executes many instructions per bit.

```
for (fd = 0; fd < maxfd; fd++) {
  if (!FD_ISSET(fd, &readfds) && !FD_ISSET(fd, &writefds))
   continue;
  if (FD_ISSET(fd, &readfds)) {
   if (fd_table[fd].read_handler) {
       fd_table[fd].read_handler(fd, fd_table[fd].read_data);
   }
  }
  if (FD_ISSET(fd, &writefds)) {
   if (fd_table[fd].write_handler) {
       fd_table[fd].write_handler(fd, fd_table[fd].write_data);
   }
  }
}
```

Figure 3. Another source-code excerpt from `comm_select()`

The bitmaps typically have only a few bits set, and so it is much more efficient to treat each of them as an array of words, then iterate over this array looking for nonzero words (those with at least one bit set). This effectively parallelizes the search, and uses fewer instructions per step, so the revised loop executes (on average) a small fraction of an instruction per bit. Once it discovers a nonzero word, a simple loop then finds the nonzero bits in this word, calculates the corresponding bitmap index, and invokes the corresponding handler.

With this change, `comm_select()` now uses only 0.54% of the nonidle CPU cycles. Its CPI increases somewhat, to 1.18, because the last change eliminated lots of "useless" instructions but still examines the same bitmaps (and so still has about the same set of memory references). In spite of this dramatic improvement, `comm_select()` still has $O(N^2)$ behavior; the size of the bitmaps it manipulates remains proportional to the request rate. However, the changes reduce the cost by such a large constant factor as to make it unlikely that the `comm_select()` would ever dominate system costs at any request rate.

Having succeeded in reducing the profiled cost of `comm_select()` *from 30% of the nonidle cycles to just 0.54%, my last task was to verify that this actually improves overall performance.*

Measuring the Results

Having succeeded in reducing the profiled cost of `comm_select()` from 30% of the nonidle cycles to just 0.54%, my last task was to verify that this actually improves overall performance.

Recall that I have samples, taken every 15 minutes, of both the request rate and the CPU utilization. On separate days, I ran trials of the original Squid proxy and the modified software; each 24-hour period yielded 96 samples. Figure 4 shows both sets of samples. Idle-time measurements are shown using circles, and user-mode CPU time measurements are shown using squares. Samples from the trial with the original software are distinguished by open marks; samples from the trial with the modified software are shown using filled marks.

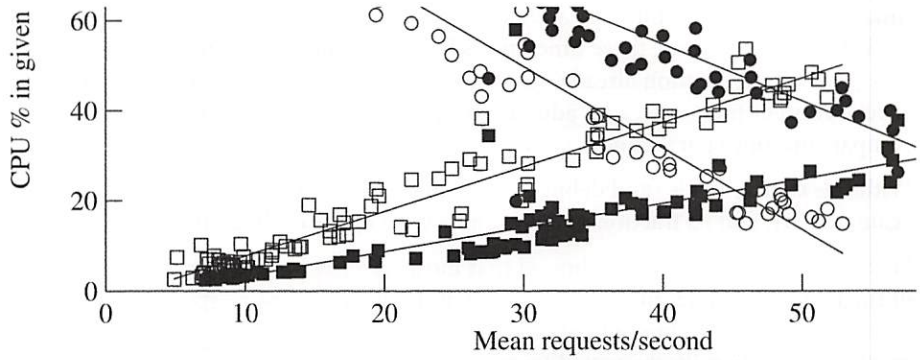


Figure 4. CPU costs as a function of request rate

Figure 4 includes lines showing linear regressions for each sample set. Table 2 summarizes these regressions. They show that the modifications make a 45% reduction in the slope of the regression for user-mode time; that is, the modified Squid expends much less user-mode CPU time per request than the original version. The slope of the regression for kernel-mode time has also improved slightly, apparently because the reduction in user-mode D-cache misses has improved the cache performance for other parts of the system. Overall, the slope of the idle-time regression has improved by 30%. In other words, for a given request rate, the modified software leaves more idle time and hence should support a higher request rate.

The X-intercepts of the idle-time regression lines imply that the modified version of Squid, on this hardware, can support a 45% higher request rate than the original. (The X-intercept is the projected point where the idle time drops to zero.) However, this projection is misleading; the system can actually sustain much higher request rates. The peak one-second rate actually logged during the trial with the original version was about 107 requests/sec, and was about 139 requests/sec with the modified version.

There's a simple reason for the discrepancy between the measured peak rates and the projected X-intercepts. When the system is lightly loaded, events arrive rarely enough that `select()` typically returns with just one ready descriptor. At increased loads, when the system has no idle time and the software falls behind, `select()` will return multiple events per call. In this region, each call to `select()` is amortized over multiple events, so its cost per event is reduced. Because `select()` and/or `comm_select()` account for the majority of the CPU-time costs, this "batching" effect allows the system

to handle increasing load even after running out of idle time. Ultimately, however, the more efficient software can handle a higher peak load, and provides less latency at lower loads, than the original version.

Squid Version	CPU mode	Slope	Correlation coefficient	X-intercept
Original	User	0.99	0.97	
Modified	User	0.54	0.80	
Original	Kernel	0.82	0.98	
Modified	Kernel	0.73	0.95	
Original	Idle	-1.81	-0.98	57.5
Modified	Idle	-1.26	-0.92	83.4

Table 2. Linear regressions of CPU costs as a function of request rate

Next Steps

The results of this case study suggest further steps to take. First, my patches to `comm_select()` will be integrated into a future release of Squid.

After application of these patches, and the kernel modifications to improve the performance of `select()` [3], the profiles still show that `select()` consumes a lot of CPU time, with user-level `malloc()` not far behind. A forthcoming release of Squid maintains its own pools of allocated memory for frequently-allocated data types, and so should put a lot less pressure on the `malloc()` functions, but it might still be worthwhile to study this cost in more detail.

Reducing the cost of `select()` while preserving its programming interface, beyond the kernel modifications we have already tried, may prove difficult. In another paper [2], Gaurav Banga, Peter Druschel, and I propose a new system call interface to replace `select()` for this kind of application; preliminary results from simulated-WAN testing suggest that this interface eliminates virtually all of the `select()`-related costs, and scales to arbitrary numbers of file descriptors.

Conclusions

I set out to discover something interesting about how Squid's CPU utilization scales with request rate, and why. By using the appropriate performance-debugging tools, I quickly discovered the cause of a significant inefficiency. After a few false starts and some additional tool use, I devised a relatively simple change to the Squid code. This eliminated most of the user-mode CPU time spent in the Squid program. More important, it increased the efficiency of the entire system, as measured by tests on a live system.

Of course, not every program has a single procedure whose performance can be improved by a factor of 55 through a simple programming change. But many programs, especially Internet servers in complex environments, have performance characteristics that are not necessarily obvious from their source code. Further, the influence of memory-system stalls is both especially critical on modern CPUs and especially difficult to discern from the source. One can gain tremendous leverage over these problems by using advanced performance tools, such as DCPI, that provide whole-system performance views and expose memory-system issues.

References

[1] Jennifer M. Anderson, Lance M. Berc, Jeffrey Dean, Sanjay Ghemawat, Monika R. Henzinger, Shun-Tak A. Leung, Richard L. Sites, Mark T. Vandevoorde, Carl A. Waldspurger, and William E. Weihl. "Continuous Profiling: Where Have All the Cycles Gone?" *ACM Transactions on Computer Systems* 14(4):357-390, November, 1997.

[2] Gaurav Banga, Peter Druschel, and Jeffrey C. Mogul. "Better operating system features for faster network servers." *Proceedings of the Workshop on Internet Server Performance,* pp. 69-79. Madison, WI, June, 1998. Available as <http://www.cs.rice.edu/~gaurav/papers/wisp98.ps>; revised version in *ACM SIGMETRICS Performance Evaluation Review* 26(3), Dec. 1998.

[3] Gaurav Banga and Jeffrey C. Mogul. "Scalable kernel performance for Internet servers under realistic loads." *Proceedings of the USENIX 1998 Annual Technical Conference,* pp. 1-12. New Orleans, LA, June, 1998.

[4] Anawat Chankhunthod, Peter B. Danzig, Chuck Neerdaels, Michael F. Schwartz, and Kurt J. Worrell. "A Hierarchical Internet Object Cache." *Proceedings of the 1996 USENIX Technical Conference,* pp. 153-163. San Diego, CA, January, 1996.

[5] Digital Equipment Corporation. DIGITAL Continuous Profiling Infrastructure Project. <`http://www.research.digital.com/SRC/dcpi/`>. You can obtain DCPI software and documentation from this URL.

[6] John Lions. Lions' Commentary on UNIX 6th Edition with Source Code. Peer-to-Peer Communications, San Jose, CA, 1996. Shows that, at least as far back as 1976, UNIX included a `prof()` system call.

[7] National Laboratory for Applied Network Research. Squid Internet Object Cache. <http://squid.nlanr.net/Squid/>. You can obtain Squid software and documentation from this URL.

using java

The Java Native Interface

In a previous *;login:* article (June 1998), I discussed the embedding of a Java Bean into an ActiveX environment. I showed that components written in Java can interoperate with components written in other languages at the "component" level, namely ActiveX components. This article continues the theme of interoperability, but with a different perspective. We will look at the Java Native Interface (JNI), which is also a very important part of the Java development infrastructure. The JNI capability permits programs written in Java to call programs written in C or C++, and also the reverse: a C or C++ program can invoke the Java Virtual Machine (JVM) and invoke methods on a Java object.

by Prithvi Rao

Prithvi Rao is the founder of Kiwilabs, which specializes in software engineering methodology and Java training. He has worked on the development of the MACH OS and a real-time version of MACH, and he holds two patents resulting from his work on mobile robots.

<prithvi+@kiwilabs.com>

This capability is mandated in several situations. For instance, in order to run Abstract Windowing ToolKit (AWT) applications, the graphics context for the application has to eventually make "native" calls to the libraries that perform the graphics operations. It is likely that these libraries are written in C (more likely than C++). Also, Java communication packages eventually make "native" calls; we can gather from this that "native" refers to either C or C++. Finally, it is likely that most organizations have significant amounts of C and C++ code that has been debugged and tested. JNI plays a vital role in supporting the use of this code.

Let's start with a summary of the main considerations of JNI, then present examples of code and a more detailed description of this capability.

Java Native Interface (JNI)

The Java Native Interface is a standard programming interface between Java and native programs. Typically, native programs are used when they cannot be written in Java (legacy code, performance considerations).

The JNI makes it possible to create, inspect, and update Java objects including arrays and strings. The JNI also permits the catching and throwing of exceptions, loads classes, obtains class information, and performs runtime type checking. It is possible using JNI to link a Java Virtual Machine (JVM) into non-Java applications.

As with any important technology, interested parties will jostle for acceptance of their ideas. Sun, with its JNI, is one of three currently trying to influence the Native Interface standard. The other two main parties who can influence the standard are Netscape with the Java Runtime Interface (JRI) and Microsoft with the Raw Native Interface (RNI). The standard is still evolving, and this article will discuss only JNI from Sun.

The advantages of a standard are that each vendor can support a large body of native code. Also, tool builders need not maintain different kinds of native method interfaces. This means that application programmers write one version of their native code. Some requirements of the evolving standard are:

- Binary compatibility so that one version of native code exists.
- Efficiency so that the overhead of the interface should be small.
- Functionality so that native methods must be able to do useful things.

It appears that JNI is becoming the standard, so it is probably a good idea to program to it.

Let's take a look at an example of how JNI can be used to call C programs from Java.

Working with Native Methods

The steps involved in using JNI are:

1) Write the Java code and define relevant methods "native."

2) Compile the Java code.

3) Create a header file that acts as a bridge between Java and native code.

4) Implement the native code.

5) Create a shared library of the native code.

JNI Example

Let's take a look at an example of how JNI can be used to call C programs from Java. (In a future article I will discuss how to do the reverse.) We will look at a package to manipulate rational numbers. The package supplies the following routines:

- `radd` - add two rational numbers
- `rsub` - subtract two rational numbers
- `rmul` - multiply two rational numbers
- `rdiv` - divide two rational numbers
- `euclid_gcd` - the greatest common divisor of two integers

We will create a Java class called Rational:

```
public class Rational
{
  public int[] r;
public Rational(int n, int d)
   {
     if (d == 0)
      throw new ArithmeticException("Rational: requires " +
                                          "non-zero denominator");
     r = new int[2];
     r[0] = n;
     r[1] = d;
   }
public static Rational RationalAdd(Rational r1, Rational r2)
{
     return new Rational(add(r1.r, r2.r));
}
public static Rational RationalSub(Rational r1, Rational r2)
{
     return new Rational(sub(r1.r, r2.r));
}
public static Rational RationalMul(Rational r1, Rational r2)
{
     return new Rational(mul(r1.r, r2.r));
}
public static Rational RationalDiv(Rational r1, Rational r2)
{
     return new Rational(div(r1.r, r2.r));
}
```

Next we define the native methods:

```
public static native int[] add(int[] r1, int[] r2);
```

This says that there is a function called "add" which is not written in Java.

```
public static native int[] sub(int[] r1, int[] r2);
public static native int[] mul(int[] r1, int[] r2);
public static native int[] div(int[] r1, int[] r2);
public static native int[] gcd(int x, int y);
```

Next we create a `toString` method. This is always a good idea:

```
public String toString()
{
  StringBuffer sb = new StringBuffer();
  sb.append(r[0]);
  sb.append('/');
  sb.append(r[1]);
  return new String(sb);
}
```

Now we need to add code to load the shared library:

```
static
{
  System.loadLibrary("rat");
}
```

This call loads the `librat.so` (Solaris) shared library so that Java programs can invoke the native calls.

Now that we have the Rational class defined we can compile the code:

```
javac Rational.java
```

This produces a .class file from which we create a header file which acts as a "bridge" between the Java and C code.

```
javah -jni Rational
```

The output of this is a file "Rational.h". The entries in this file must not be edited, because the JVM uses it to interpret the parameters being passed between Java and the native code.

For our native function "add" the entry in this file looks like this:

```
/*
 * Class:                     Rational
 * Method:                    add
 * Signature: ([I[I)[I
 */
JNIEXPORT jintArray JNICALL Java_Rational_add
    (JNIEnv *, jclass, jintArray, jintArray);
```

Every native function will have a similar entry based on its signature. We will discuss the details in a subsequent article.

Next we write the C code that implements the native functions. We include only the "add" example here. Readers can send mail to <prithvi+@kiwilabs.com> for a full code example.

It is entirely up to the programmer to guarantee that the native call is thread-safe.

```
/*
The interface between java and C for the rational number package
*/

#include <jni.h>
#include "Rational.h"
#include <stdio.h>

int *radd(int *, int *);

JNIEXPORT jintArray JNICALL
Java_Rational_add(JNIEnv *jenv, jclass jc, jintArray jr1,
          jintArray jr2)
{
jint *r;
jintArray jr;
jsize rlen = (*jenv)->GetArrayLength(jenv, jr1);
jint *r1arr = (*jenv)->GetIntArrayElements(jenv, jr1, 0);
jint *r2arr = (*jenv)->GetIntArrayElements(jenv, jr2, 0);

(*jenv)->ReleaseIntArrayElements(jenv, jr1, r1arr, 0);
(*jenv)->ReleaseIntArrayElements(jenv, jr2, r2arr, 0);

r = (jint *) radd((int *) r1arr, (int *) r2arr);
jr =  (*jenv)->NewIntArray(jenv, rlen);
(*jenv)->SetIntArrayRegion(jenv, jr, 0, rlen, r);

return jr;
}
```

The above code example is the native code implementation of "add". In subsequent articles I will cover in more detail the writing of native functions. For now it is important to note that native code includes the "jni.h" and "Rational.h" include files. These two files are crucial because they form the "bridge" which permits the Java native programs to pass parameters between them.

JNI and Threading

Since Java is a multithreaded language, several threads can call a native method concurrently. However, it is possible that a native method might be suspended in the middle of its operation when a second thread calls it. It is entirely up to the programmer to guarantee that the native call is thread-safe. One way of doing this is to declare the native thread as "synchronized" or implement some other strategy within the native method to ensure correct concurrent data manipulation.

Another consideration of multithreading is that the JNIEnv pointer must not be passed across threads. This is because the internal structure to which it points is allocated on a per-thread basis and so contains information that makes sense only in that particular thread.

JNI and Java Exceptions

With JNI, Java exceptions can be thrown, caught, printed, and rethrown just as they are inside a Java program. But it's up to the programmer to call dedicated JNI functions to deal with exceptions. Some JNI functions for exception handling are:

- `Throw()`: Throws an existing exception object. Used in native methods to rethrow an exception.
- `ThrowNew()`: Generates a new exception object and throws it.
- `ExceptionOccurred()`: Determines if an exception was thrown and not yet cleared.
- `ExceptionDescribe()`: Prints an exception and the stack trace.

- `ExceptionClear()`: Clears a pending exception.
- `FatalError()`: Raises a fatal exception.

Of these exceptions, it is not possible to ignore `ExceptionOccurred()` and `ExceptionClear()`. After exceptions are handled by `ExceptionOccurred()` it is necessary to clear them using `ExceptionClear()` to avoid unpredictable results (especially if a call is made to a JNI function while an exception is pending).

Conclusion

We have looked at the use of JNI for calling native methods written in C from a Java program. Clearly, this is a very important consideration in Java being a credible language; Java could not be taken seriously were this not possible. It is equally important to be able to invoke Java methods from native code, and this is also possible.

A strength of Java is that not only is native code integration possible, but it is simple and straightforward with all the tools available as part of the JDK distribution. However, it is still necessary to understand the use of JNI within the larger context of multithreading and exception handling, which have important implications when you are working with native interfaces.

The Great Certification Debate

This interview with Barb Dijker was conducted by Rob Kolstad just prior to LISA and its Great Certification Debate. Barb has been active in getting the certification effort off the ground.

Interview with Barb Dijker

Rob: How did you become involved in the certification effort?

Barb: The same way I became a system administrator. It's something that appears to need doing, someone needs to do it, and it wasn't getting done. I didn't start this, nor did I push it through the SAGE Executive. This is a significant undertaking with broad implications. I'm one of four on the SAGE certification subcommittee. Another 48 SAGE members are on an advisory council – which will hopefully keep us from screwing up.

Rob: Are you personally for or against certification?

Barb: Personally I don't have strong inclinations one way or the other. The reason is that I don't think we really have enough data yet to substantiate arguments on either side of the issue. That's why this effort is important. The initial goal is to get the data so we can make a decision.

Rob: And what is the final goal of SAGE certification?

Barb: Once a certification program is developed, its purpose would be to provide an objective means of skill assessment. It would be a guide for new system administrators to develop their skills and a means for hiring managers to wade through resumes.

Rob: Does anyone really want that?

Barb: Employers eat it up. Those looking to get into the high-salary world of system administration would have a road map to do it. I'm serious. To a PC repair person, system administration is "high-salary" even if certification does water down salaries as doomsayers claim it will. Anyone with good problem-solving skills can learn to be a system administrator. We need more system administrators. How else are we going to grow them?

Rob: Shouldn't we be helping universities build curricula instead?

Barb: In addition to, but not instead. Getting a degree in computer science, even if there were a minor in system administration, isn't going to necessarily teach you everything you need to know about doing it on the job. Nor does it tell a potential employer how much of that curriculum soaked into your brain and contributed to your actual skill level. If higher education were the only answer, there would be no certification programs anywhere.

Rob: How do you implement it?

Barb: SAGE has never done anything this big. So implementation is as much of the problem as determining the required skill sets and developing methods to assess them. However, there are many professional certification-program development and execution companies that can help us. SAGE isn't the first professional or trade association to want to provide a certification program for its members. Think of teachers, nurses, cops, etc. All of them get "professional" outside help to develop their programs. As system administrators, we want our users to recognize when they should call in a "professional" rather than taking the do-it-yourself approach. By the same token, we need to recognize that developing a certification program is completely outside of our skill set.

Rob: This sounds really expensive. How can it be worked out so it's affordable to those being certified?

Barb: It may turn out that it is cost-prohibitive. We don't have all the data. At this point, we aren't developing a certification program yet. We are only doing a feasibility study to justify a recommendation of whether to develop and implement a certification program or not. Part of the criteria for the recommendation is that the program must be affordable. If it isn't affordable to the participants, then it doesn't have enough value.

On the other hand, SAGE isn't doing this to make a profit like a company. So the program only needs to cover the cost of implementation and maintenance.

Rob: Is everyone in favor of this?

Barb: That's impossible to say. Everyone on the SAGE Executive was in favor of this effort – or at least not opposed to it. When SAGE conducted a survey last year, there was 2:1 support. However, we know that the survey results do not statistically represent the SAGE membership or the greater system-administrator community. It's just an interesting datapoint – a conversation starter. And boy did it stir up conversation. When the SAGE Executive first announced the plan to start down this road back in February, the <sage-members> mailing list was flooded with a heated debate. A certification program isn't any good unless it is accepted by a reasonable majority of the membership. That's why a major aspect of this effort is to open constructive dialogue about certification. A Web site, <http://www.usenix.org/sage/cert/>, is online for information and feedback and debates, and BOFs are being conducted.

Initially the feedback was negative. This was mostly based on fear that SAGE would recreate MSCE – which is generally viewed by SAGE members as an expensive waste of time – or that SAGE would mandate certification for membership or try to certify senior sysadmins. Since those fears have been dispelled, most of the feedback has been a very positive and "it's about time" sort of thing.

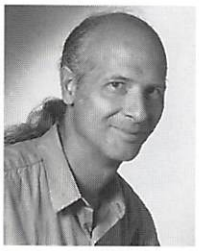
Rob: Do I need to keep getting recertified every year or two or keep getting add-on credits or something?

Barb: That has not yet been determined. The initial focus is on "core competency." That basically means certification for a solid entry-level system administrator. However, another goal is to design the program so that it is extensible. So in the future there might be a "postmaster" certification, for example. A moving target like that would need recertification, or at the very least a date stamp, to be meaningful.

Rob: Thanks!

"A certification program isn't any good unless it is accepted by a reasonable majority of the membership. That's why a major aspect of this effort is to open constructive dialogue about certification."

musings



by Rik Farrow

Rik Farrow provides UNIX and Internet security consulting and training. He is the author of *UNIX System Security* and *System Administrator's Guide to System V.*

<rik@spirit.com>

I just got back from the LISA conference in Boston. Boston was okay: not as cold as I had feared, plenty of good restaurants, and lots of people did show up. I think it was one of the largest LISA conferences ever, actually. While standing facing a wall, as men often do, an exhilarated attendee told me that he got more from standing in the halls talking with people than at many other conferences he had attended. I can't say that I was surprised.

Of course, there were also the traditional activities, such as listening to papers or invited talks, the LISA quiz show, and drinking beer (that's why we were standing at the wall).

Eric Allman, author of sendmail and a genuine USENIX hero, gave the keynote. Eric has decided that it was high time he made some money from sendmail. He once told me that he wished that he had a dime for every copy of sendmail that was in use. He may do better than that for *some* of the copies out there, once the commercial version of sendmail appears. But sendmail will remain Open Source (a trademark of the Open Source Initiative; see <www.opensource.org>).

There was a time when I would have defined the USENIX Association as the largest organization founded on the idea of free software. The heroes of USENIX, like Eric, produced useful tools or programs and made not only the compiled programs but the source to the programs available as well. Revealing the source has several side effects, not the least of which being what happens when many other programmers start looking over your code. This is not for the faint of heart, or the truly arrogant either. Instead, it is both a humbling and a useful experience. People will find fault with either your coding or your designs (and often both), but in the end the code gets better.

Opening Remarks

Eric began the keynote by describing the history of software as a three-act play. The first act included mainframes and minicomputers. During the question-and-answer session that followed, Peter Salus pointed out that open source had its roots in SHARE, when '50s (and later) mainframe programmers shared source code on punched cards. But that era is best described as closed and proprietary, with large expensive computers in glass rooms (think of glass display cases) and a priesthood to minister to their needs. Digital Equipment's PDP series did make computing more accessible (computers in the tens of thousands of dollars instead of millions), but the operating systems and software were still closed.

Act 2 in this chronology was the advent of first the homebrew, then the personal computer. Both the homebrew community and the early Apple computers (I and II) sponsored an "extend and return" mentality. Programmers would generally share source and communicate any extensions or changes made back to the original author. Note that Bill Gates began Microsoft by *not* doing this. He and a few programmers created a proprietary version of what had been a free version of BASIC and began selling it. Their extensions were never returned to the community, which was the original reason for the animosity toward Microsoft.

Act 3 opens with the beginning of virtual open source – UNIX and the Berkeley Software Distribution, or BSD. The UNIX system was actually proprietary, but because of the actions of AT&T, many schools and universities had access to the source code. BSD included many extensions to the standard UNIX distribution and was controlled

by the Computer Science Research Group (CSRG) at the University of California, Berkeley campus.

Bill Joy, better known as one of the four founders of Sun Microsystems, crafted the early releases of BSD. Not only did he contribute code (`vi` and `csh`, for example), he also decided what software would be included in each release.

Eric pointed out that Samba, FreeBSD, NetBSD, OpenBSD, Linux, Apache, BIND, and sendmail all share a similar feature – that of a central, controlling person or group. Without some form of release management – deciding what features and patches will make up a release – software tends to fragment. Core teams watch over the BSDs, Linux, Apache, Samba, and BIND, while Eric watches over sendmail. In this, Eric disagrees with Eric Raymond's "The Bazaar and the Cathedral" paper (<http://www.tuxedo.org/~esr/writings/cathedral-bazaar/>), which describes a somewhat different model of open-source creation.

Eric pointed out that Samba, FreeBSD, NetBSD, OpenBSD, Linux, Apache, BIND, and sendmail all share a similar feature – that of a central, controlling person or group.

Money

Open Source also means the birth of hybrid organizations, such as Red Hat or Eric's Sendmail, Inc. These organizations still give away the source but make money while doing so. Red Hat makes money by selling CDs and documentation and by providing handholding. Sendmail, Inc. will make money by selling extensions, such as an easy-to-use configuration GUI, as well as commercial support.

If you visit the Open Source Web page, you will quickly discover that the difference between "free software" and open source is largely one of marketing – but not totally. Marketing, because you cannot sell "free software." And not totally different, because open-source software will be freely available, although covered by license agreements. A lot of the Open Source Web site is devoted to explaining how open-source software can be a good thing for programmers.

And big business as well. IBM Research released Jikes, a Java compiler, as open source the first week of December. Earlier in this year, IBM also announced that it would offer support for the Apache Web server. If IBM can make money supporting software that they did not create and do not own, why shouldn't it? Service and support have always been a large part of computer companies' income. While IBM's support helps to legitimize open-source software, it also places IBM in competition with non-IBM programmers and consultants. At this point, I think IBM's recognition is more important than possible competition.

Speaking of competition, one of the LISA invited talks set the authors or representatives of four MTA (Mail Transport Agents) head-to-head in a panel. Besides Eric Allman, there was Wietse Venema, who had just completed Postfix (previously Vmailer; see <www.ibm.com/alphaworks>), Dan Bernstein, author of qmail (<www.qmail.org>), and a representative of the University of Cambridge, author of exim. Interestingly to me, all three replacements for sendmail claim better security as one of their main goals. But sendmail has had a remarkable security track record *recently,* leaving security as perhaps a poor method for choosing an MTA. Visit the Web sites (including <www.sendmail.org>) and choose your own favorite – if you haven't already done so.

Dan Klein repeated his talk at LISA on the "Succumbing to the Dark Side of the Force: The Internet as Seen from an Adult Website." Although you might have expected Dan's talk – which does include lots of useful Web-server lore and little about the other side of the business – to have drawn the entire conference attendance, the papers track,

I quickly learned how wrong I was to assume that a head-mounted display should be much less power hungry and easier to support than a large notebook display.

including a talk by Chris Page about configuring database servers, drew a large crowd as well.

The Borg

I had hoped that while I was in Boston I could slip away from the conference and meet with some of the wearable-computing folk, occasionally known as the borg, over at the MIT Media Lab. I had not counted upon the end of the semester, as well as a thesis defense, occurring at the same time. I did get to meet with Dave Kaplowitz (Dkap), now at Tufts University, who showed me around the Media Lab Friday afternoon. You may have seen Dkap at the last LISA conference, in San Diego, and he was also in attendance in Boston. His rig includes a Private Eye display, a Twiddler for input, a box containing an Intel 486 processor and hard drive, and a fairly heavy set of lead-acid batteries.

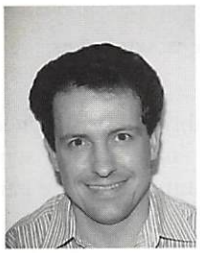
When we met at MIT, Dkap had left his wearable at Tufts. There had been a security incident, and he had left his wearable behind working on a Tripwire check of the invaded system. The lead-acid batteries, although bulky, do provide all-day power and make a useful system for Dkap in his system-administration duties. It did seem strange to see him without his wearable.

I also had a chance to talk with Steve Schwartz, a research scientist hired specifically to work on wearables at the Media Lab. Steve has worked with head-mounted displays (HMDs) for many years, starting with work with Lucas Films in the '80s. Steve provided me with a rapid-fire history of HMDs, and I quickly learned how wrong I was to assume that an HMD should be much less power hungry and easier to support than a large notebook display. HMDs can be less power hungry, but getting them to work requires special hardware – they do not work directly from VGA output, any more than the new flatscreen display monitors do. What's worse, companies that have focused on producing HMDs have had a bad habit of either going out of business (as Private Eye has) or focusing on other products.

As I wandered around the Media Lab, I also learned that the wearables group (<www.mit.edu/wearables>) is spread throughout the building and organization. For most people in the Media Lab, a wearable computer is not a goal, but rather a tool for exploring new applications made feasible by the wearable. Software agents that adapt to what you are doing as well as your past behavior. Effective computing, getting computers to recognize and respond to human moods. Not being strange, borglike humans.

I plan on revisiting the topic of wearables in the not-too-distant future. I am astonished by the number of people in the USENIX community who are already using wearables -- mostly PalmPilots, with a scattering of other types of what are today called handhelds. These computers are worn until they are needed, then they are opened, turned on, then put back into a pocket until needed again. Not quite true wearables, but getting close. And the age of true wearable computing is not quite here yet, either.

standards reports



edited by Nicholas M. Stoughton

USENIX Standards Liaison

<nick@usenix.org>

The Single European Currency

Finnbarr P. Murphy <fpm@zk3.dec.com>

Abstract

By now we are aware of the Year 2000 problem. However, enterprises doing business in or with European counties have to contend with another major software issue, the introduction in January 1999 of a new single European currency called the euro. This article is a brief introduction to the euro and the impact that its introduction will have on operating systems and applications.

Introduction

In 1998, the third stage of the Economic and Monetary Union (EMU) was launched with the establishment of a European Central Bank (ECB) in Frankfurt, Germany, to manage the introduction of a new single pan-European currency. The name chosen for the basic denomination of this currency was the euro.

The introduction of the euro will be one of the major economic events in the next decade. A common currency will contribute significantly to the further economic integration of the participating countries. It will increase market transparency by making prices more easily comparable. Pan-European trade will become more attractive as trade and investment will no longer be exposed to exchange-rate risks, and the costs associated with currency conversion will be eliminated. Eleven countries qualified for initial membership in the EMU: Austria, Belgium, Finland, France, Germany, Ireland, Italy, Luxembourg, Portugal, Spain, and the Netherlands.

On January 1, 1999, the rate of conversion between the euro and the currencies of the eleven participating countries will be irrevocably fixed, and the ECB will begin the administration of a single monetary policy. On this date, the euro, in non-cash form, will become legal tender in these countries. This means that the euro will be usable for non-cash transactions such as checks and credit transfers.

On January 1, 2002, new euro banknotes and coins will be put into circulation. Over a period of time, determined by each participating country, notes and coinage in the national currency will be withdrawn from circulation. By July 1, 2002, euro notes and coins will be the only legal tender in the participating countries.

Currency Symbol

The European Commission has introduced a new currency symbol to represent the new currency and has registered the currency code EUR with the ISO 4217 (Currency Codes) Maintenance Agency as representing this currency.

The new currency symbol is essentially a three-quarters circle with two horizontal crossbars, as shown below:

€

Support for the euro currency symbol in computer systems and applications means the ability to input, display, print, store, and retrieve the symbol. This requires a small number of changes to operating systems and various international standards.

The Unicode Standard and ISO/IEC 10646-1:1993 (UCS) included a symbol called the Euro-currency sign in the block of currency symbols at code posi-

The following Reports are published in this column:

Our Standards Report Editor, **Nick Stoughton**, welcomes dialogue between this column and you, the readers. Please send your comments to: <nick@usenix.org>

tion 0x20a0, and a corresponding glyph "CE" consisting of an interleaved "C" and "E", with the "E" being lower and more to the right than the "C". The meaning of this symbol was recently clarified and declared to represent the ECU (European Currency Unit).The ECU is an existing, mainly paperless, currency used within the European Union. Its value is based on a basket of national currencies. One ECU will be equivalent to one euro. Upon the introduction of the euro, use of the ECU will be phased out.

In order to represent the euro currency, a new symbol called "EURO SIGN" was added to ISO/IEC 10646-1 in the block of currency symbols at code position 0x20ac, and a new glyph representing this symbol was registered in the ISO/IEC 10036 Glyph Registry.

Images of the EUROCURRENCY SIGN and EURO SIGN glyphs, and the new notes and coinage may be viewed at <http://www.euro.eu.int/euro/html/entry.html>.

The Final Committee Draft ISO 8859-15 (Latin9) standard (nicknamed Latin0 because it updates Latin1 with some forgotten French and Finnish characters) replaces the code-point for the generic international currency symbol (0xa4) with the euro currency symbol.

The HTML 4.0 specification defines the entity "€" ("€") as the euro currency symbol.

Support for the euro is planned in the next major release (V1.2) of the Java Development Kit (JDK). Before then, support will be provided in a maintenance release (V1.1.7) of the JDK because of the immediate need for such support. The specification of the `java.lang.Character` class was updated to include the euro currency symbol.

The X11R6.4 Sample Implementation included support for the euro currency symbol in Public Patch Number 2. The € symbol is defined in <`X11/keysymdef.h`>. However, many of the fonts in the Sample Implementation use ISO8859-1 encoding, which does not include support for the euro currency symbol.

A large number of font vendors have produced new fonts which include the € glyph. For example, Adobe Systems has developed a number of Type 1 Postscript fonts. These are available for free download at <http://www.adobe.com/type/eurofont.html>.

Most operating-system vendors are planning to include at least basic support for the new currency in their next major release. Typically this support will include the following:

- A set of UTF-8 locales which will include support for the € symbol
- Extensions to the X Window System font library to enable display of these locales
- Modification of locale database and resource files to support the new UTF-8 locales
- Input method and printing support for the € symbol as required

The complete set of UTF-8 locales will probably not be supplied in the first release containing support for the euro currency symbol. Expect to see UTF-8 locales for those countries that are part of the European Monetary Union first.

Impact on Applications

The impact of the introduction of the euro currency is essentially a software issue – only systems that process financial information in one of the participating national currencies are affected by the euro changeover.

For organizations doing business in any of the participating countries, modifying applications to accommodate the euro is of the highest priority – probably even more than the Year 2000 problem because of the January 1, 1999, introduction date. The Year 2000 problem is basically a technical problem, whereas the euro currency changeover requires additional functionality in applications.

Where the impact of the euro currency introduction will be felt most is in large multinational companies, stock and bond markets, and all levels of the financial markets. This will affect non-European banks, financial institutions, and businesses also.

Examples of applications that are affected include:

- Accounting systems
- Invoicing and billing systems
- Payroll systems
- Fixed asset systems
- Financial planning systems
- Enterprise resource planning systems
- Treasury management systems.

Generic software such as Microsoft Office and Lotus SmartSuite will also have to be upgraded. Many spreadsheets dealing with financial information will have to be redesigned.

Conversion between the euro and a participating national currency will cause a rounding error and a loss of precision. Specific legal requirements for expressing amounts in euro and for converting between national currencies have been established, and applications will have to be modified to ensure that these requirements are fully implemented. Of particular significance are rounding errors caused by cumulative data operations. Conversion between two national currencies must be via the euro. The use of inverse exchange rates to convert amounts

between a national currency and the euro is not permitted. Many existing applications convert currencies using cross rates and inverse rates. Such applications will require extensive modification.

Applications that were designed to work with a national currency with no decimals will need to expand the currency field to work with decimals, since the euro currency includes cents (100 cents to the euro). Such countries include Spain, Belgium, and Italy.

It may be difficult to display the euro and a national currency simultaneously, as required during the transition period (January 1999 to mid-2002), when both the euro and national currencies are valid legal tender (dual currency), because of a shortage of screen and printed report space. Some applications may require a major redesign to comply with this requirement.

Many applications use some sort of threshold value to define various actions. For example, a business may have one shipping charge for orders under $500.00 and a different charge for orders over $500.00, with the application automatically calculating the shipping charge. Such thresholds will have to be identified and changed. Other common thresholds used in applications include authorization levels, data validity checking, and exception report generation.

It may not be possible to change all applications over to the euro at the same time. This means that applications will have to communicate with applications that are not euro-compliant. Many enterprises have links to systems belonging to other enterprises. In this case, the changeover to the euro requires careful coordination between the various enterprises.

At some time, enterprises will have to convert over to the euro currency completely. When this occurs, a considerable amount of historical data will probably also have to be converted from the national currency to the euro.

Keyboards

Keyboard standards distinguish three main levels of functionality:

- Level 1 – Press the "m" key to produce the character "m".
- Level 2 – Press the Shift key and the "m" key simultaneously to produce the uppercase character "M".
- Level 3 – Press the AltGr key and the "e" key simultaneously to produce the € symbol.

The European Commission is determined that the euro currency symbol € will have a prominent, i.e. visible and easy to use, placement on user's keyboards, and has issued two proposals.

The short-term proposal is for all Latin and Greek alphabet-based keyboards to place the euro currency symbol on the "E" keycap (key D 03 on Level 3). Simultaneously pressing the "AltGr" (The Right Alt key) and "E" keys generates the euro currency symbol. Many existing European keyboards use this type of engraving. This solution is common to all major types of keyboards, is simple to implement, and is easy to remember because of the association of E with the euro. However, this proposal causes problems for written Irish (Gaeilge) and some other languages. A number of vendors are using AltGr+4 for Irish and United Kingdom keyboards, AltGr+5 for Greek Latin and Hebrew keyboards, and AltGr+u for Hungarian and Polish keyboards. On keyboards with no AltGr key, the euro currency symbol may be produced by the sequence Ctrl+Alt+E.

A new draft of the ISO/IEC 9995-3 keyboard standard is being balloted. This includes the € in the common secondary layout and thus provides for the placement of the € in those keyboards where the primary layout does not include the €.

The long-term proposal is to introduce a new keycap which will be placed in the same location on all major keyboard layouts. This would be a level 1 key. The proposed location is under the Carriage Return key, to the left or right of the Shift key located right under the Carriage Return key. A number of vendors have already introduced keyboards conforming to this proposal.

Printers

The way the euro currency symbol is handled by a printer will depend upon the operating system and the particular printer capabilities. Since the majority of existing printers do not have built-in support for the euro currency symbol (i.e., in their resident fonts) most operating system vendors will deploy a software approach using a downloadable font. In essence, the euro currency symbol will be treated as a small graphic in a font which can be downloaded to the printer whenever the euro currency symbol is to be printed. Printing fonts as graphics instead of using a resident font may result in slower printing.

Many font vendors have made the new euro currency symbol available either as a standalone glyph which precisely matches the definition of the symbol as published by the European Commission, or as a glyph that is built into a typeface library and which matches the characteristics of a specific font. This arrangement gives users the choice of using either the precise symbol or a modified symbol which matches their favorite font.

By mid-1999, expect most new printers to have support for the euro currency symbol built into their resident fonts.

Further Information

The following documents are available for download at <http://www.ipso.cec.be/y2keuro/euroit.htm>:

- Preparing Financial Information Systems for the Euro, 12/15/97, European Commission, Brussels

- Recommendation for the Placement of the Euro Sign on Computer Keyboards and Similar Information Processing Equipment, V1.5 (6/19/1998), European Commission, Brussels.
- The Introduction of the Euro and the Rounding of Currency Amounts, Europaper N.22.
- Preparing Information Systems for the Euro, 9/25/97, European Commission, Brussels

.

POSIX.1h SRASS and POSIX.1m Checkpoint/Restart

Helmut Roth <hroth@nswc.navy.mil> *reports on the October 1998 meeting in San Diego, CA.*

The POSIX.1h Services for Reliable, Available and Serviceable Systems (SRASS) and the POSIX.1m Checkpoint/Restart working group met in San Diego, California, in late October 1998. The SRASS working group is in the process of developing a set of APIs for fault management and serviceability applications. The goal of the SRASS Working Group is to support fault-tolerant systems, serviceable systems, reliable systems, and highly available systems in a portable way. Wherever feasible, POSIX.1h needs to be useful for general applications too, such as distributed parallel database transaction systems and safety-related systems.

Checkpoint/Restart allows an application to save the entire state of the machine along with the operating system and the process activities so that in the event that something goes wrong, a saved backup state can be brought on line quickly. A ballot group has been formed, and the Checkpoint/Restart API should go out for ballot in November 1998. Some minor corrections to the draft to support a more rigid set of rules for use of the namespace and other backward-compatibility issues are being added. It is intended to be out to the ballot group in time to ballot before the next meeting in January 1999.

One part of the SRASS draft deals with logging APIs. These are aimed at allowing an application to log application-specific events and system events to a system log and allow for the subsequent processing of those events. Fault-management applications can use this API to register for the notification of events as they enter the system log. An example of an event could be where some limit has been exceeded. Events can have a severity associated with them. Event notification can provide a way to react proactively and initiate steps to prevent a subsequent system failure. In addition to these logging APIs, the *de facto* standard `syslog` interfaces have been added to support backward compatibility.

Another feature of the SRASS proposed interfaces is a single core-dump-control API. This is intended to enable an application to specify the location of a file to which a core dump will occur in the event of an abnormal termination.

A shutdown/reboot API has been included in this draft. This includes options such as fast shutdown and graceful shutdown, and features such as rebooting with optional scripts. The configuration-space-management API is intended to provide a portable method of traversing a system's configuration space, and for manipulating the data content of nodes in that configuration space. This API will provide a fault-management application access to underlying system configuration information and the means to direct reconfiguration of the system. The most recent changes in this area have been to move from a tree traversal to a directed graph.

The working group has approved the current SRASS draft 4.0 with changes to go out for ballot in December 1998; an extra meeting is planned at DISA in early December 1998 to verify that the latest changes have been incorporated correctly. If you are interested in helping support fault management (including serviceability and fault-tolerance aspects of systems), just get in touch with Helmut Roth at <hroth@nswc.navy.mil> or Dr. Arun Chandra at <achandra@vnet.ibm.com>. To subscribe to the SRASS mail list, send a message to <srass-request@pasc.org> and ask to be included.

the bookworm

by Peter H. Salus

Peter H. Salus is a member of ACM, the Early English Text Society, and the Trollope Society, and is a life member of the American Oriental Society. He has held no regular job in the past lustrum. He owns neither a dog nor a cat.

<peter@pedant.com>

Beginners

A few years ago, there was a run of "introductory" UNIX books: Harley Hahn's, Graham Glass's, the new edition of the Nutshell book. Then there were the two "Dummies" books by John Levine and Margy Young (better than one might expect). McMullen's *UNIX User's Interactive Handbook* puts these in the shade. It both permits the learner to access a training Web site, and also contains written exercises and on-screen lessons throughout. Chapter 8 on Regular Expressions and Chapter 9 on vi are very good indeed; if there's any weakness, it's the lack of discussion of shells. But perhaps that's something we don't want learners to know. McMullen has written a splendid work that takes the user from logging in and changing a password to customizing X. It's a keeper.

Spam

Schwartz and Garfinkel (*Stopping Spam*) have produced a superb introduction to the problem of spam and the possible solutions.

In nearly every form, spam is pervasive. My email today was just over 180 messages; 42 of them were spam. I received 11 pieces of US mail today; eight were unsolicited non-first class. Spam is intrusive and offensive. But while catalogs, solicitations for donations, letters telling me that I "may already have won!", and supermarket fliers may stuff my mailbox and burden my letter carrier, they do not cause service halts. In a period when ISPs have become more and more conscious of Quality of Service, the unwanted burdens of spam are truly onerous.

Schwartz and Garfinkel have produced a well-written, entertaining book that contains a lot of information, including a solid chapter on blocking spam. (The section on sendmail 8.9 and the one on the `getspam` Perl script are particularly good.)

Unfortunately, here and there Schwartz and Garfinkel wax excessively pious. Thus, on page 159 they credit a UDP (=Usenet Death Penalty) with causing a policy change at UUNET. Careful inquiry reveals that there was no change in UUNET's policy, but that there was publication of what the policy was, and that this wasn't in response to the UDP, anyway. The assertions concerning both CompuServe (now part of AOL) and Netcom are suspect, too.

The detailing of the "Green Card" (Canter and Siegel) scam and of the Spam King (Jeff Slayton) and his transgressions are extremely well done. I wish that the authors had discussed the Zilker suit (Austin, TX), which is not even mentioned. However, the chapter on legal and legislative action is very good.

All the URLs I tested worked. This is remarkable, as I frequently hit sites that have moved, vanished, or never were.

Tcl/Tk

Clif Flynt has produced a different sort of Tcl/Tk book (*Tcl/Tk for Real Programmers*), and I hope it sells a lot of copies. Unlike a book on TCP/IP I have left unreviewed, which came with a CD-ROM for "Windows 3.1 and higher" and "16MB RAM and 2MB" memory, as well as PowerPoint, Flynt's CD-ROM requires a 386 or better, "Windows 95 or NT, Mac OS7 or 8, or UNIX" and "4MB RAM or greater." Let's try to guess who's aiming at a wider audience.

Flynt writes lucidly and manages to soar above the level of *Tcl for Dummies,* while shying from the depth found in the

Books reviewed in this column:

John McMullen
UNIX User's Interactive Handbook
Upper Saddle River, NJ: Prentice Hall, 1999. Pp. 598. ISBN 0-13-099820-6.

Alan Schwartz and Simson Garfinkel
Stopping Spam
Sebastopol, CA: O'Reilly & Associates, 1998. Pp. 191. ISBN 1-56592-388-X.

Clif Flynt
Tcl/Tk for Real Programmers
San Diego, CA: Academic Press, 1999. Pp. 698. ISBN 0-12-261205-1.

James O. Coplien
Multi-Paradigm Design for C++
Reading, MA: Addison-Wesley, 1999. Pp. 280. ISBN 0-201-82467-1.

Norman Shakespeare
Year 2000 in a Nutshell
Sebastopol, CA: O'Reilly & Associates, 1998. Pp. 314. ISBN 1-56592-421-5.

Cem Kaner and David Pels
Bad Software
New York: Wiley, 1998. Pp. 365. ISBN 0-471-31826-4.

Natalie Giroux and Sudhakar Ganti
Quality of Service in ATM Networks
Upper Saddle River, NJ: Prentice Hall, 1999. Pp. 252. ISBN 0-13-095387-3.

Yves Lepage and Paul Iarrera
UNIX System Administrator's Bible
Foster City, CA: IDG Books, 1998. Pp. 628. ISBN 0-7645-3162-X.

David S. Bennahum
Extra Life: Coming of Age in Cyberspace
New York: Basic Books, 1998. Pp. 238. ISBN 0-465-01235-3.

O'Reilly or Addison-Wesley books. This really is for programmers and assumes at least some familiarity with C. I especially liked the sections on regular expressions and on exec. If you do scripting and/or gluing, this book will be a treat.

C++

C++ supports classes, over-loaded functions, templates, modules, procedural programs, and more. Before I read Coplien's lucid and succinct essay (*Multi-Paradigm Design for C++*), I had never thought about multi-paradigm design. Programmers reading this book will see just how they can combine multiple paradigms into their application development. Read together with Coplien's other books on advanced C++ and on pattern languages, one can imagine the shape of C/C++ programming in the future.

Y2K

After nearly a decade of jaw-jaw-jaw, there appears to have been a general awakening to the problems of two-digit years over the past 18 months. Shakespeare's *Year 2000 in a Nutshell* is a welcome addition to the verbiage. Like so many of the O'Reilly books, it is narrowly focussed and thorough. Nearly a third of the book is composed of a COBOL quick reference. For those of you involved in the nitty gritty (as opposed to the politics), this book will be a necessity. The first chapter should be valuable to those of you whose companies are unwilling to spend the money to rectify the problem.

QoS

Quality of Service (QoS) has become increasingly important. I reviewed Ferguson and Huston's slim book last June; I mentioned QoS with regard to spam, above. But it's necessary to understand that Internet efficiency is but the most obvious domain of QoS. In 1995, the Better Business Bureau received more complaints about computers than about car dealers. In terms of placement, computers (including software) has climbed from 20th place in 1994, to eighth in 1995, to seventh in 1996 (the last year for which I have data). This is clearly a function of the size of the potential usership as well as of the reckless distribution of insufficiently tested software.

Kaner and Pels have written a book (*Bad Software*) designed for the irate and frustrated software purchaser. I enjoyed the book and hope that it will empower folks to put a lot of pressure on the shoddy suppliers.

Giroux and Ganti have turned out a very different sort of book (*Quality of Service in ATM Networks*). Where Kercheval's book barely mentions QoS and Ferguson and Huston devoted a chapter to it, Giroux and Ganti have lavished over 200 pages on the topic. They've done a good job, too. As more and more commerce is transacted via the Internet, traffic management in general will become vital; these two authors from Newbridge Networks provide a good example of what must be done. The only shortcomings I noted were the lack of discussion of dropped packets and of latency, which certainly influence QoS.

Sys Admin

For years, Nemeth et al. and Frisch have dominated the sys admin library. A new book by Lepage and Iarrera (*UNIX System Administrator's Bible*) has come to (in some way) challenge this dominance. The best thing about Lepage and Iarrera's work is that it is simply presented. It also comes with a CD-ROM of FreeBSD 2.2.5. I admit that I wrote two of the chapters here (21 and 23) as well as part of the Introduction, so I'm not lavishing the space on the book it would otherwise deserve, but this is an entremely useful book. Well over a decade ago, I remarked to Lou Katz that I didn't want to be a system administrator. He remarked that if you ran UNIX, you'd be forced to be. This is still true, both for UNIX and for Linux.

Cyberlife

Books like David Bennahum's *Extra Life: Coming of Age in Cyberspace* make me feel very old: I took FORTRAN a decade before he was born! But this brief biography of growing up in the period of Pong and Space Invaders really made me think about the history I have written about, as well as those born into it. Bennahum writes lightly and well, and I enjoyed the book tremendously. At times funny, at others touching, Bennahum has produced a nontechnical, yet highly worthwhile, memoir.

book reviews

John Vacca

Intranet Security

Charles River Media, 1997. ISBN 1-886-80158-8. Pp. 506. CD included. $49.95.

Reviewed by Terry Rooker

<trooker@CapAccess.org>

Intranets being a hot topic, a book on their security is worthwhile. Of course, intranets are just Internet technology used for a new purpose, so it seems that there shouldn't be that much difference. As it turns out, using that technology for new purposes does change some of the assumptions, and you cannot simply apply security precautions developed for the Internet to the new application. Unfortunately, that is the weakness of this book. It rehashes all of the security techniques we've seen before and applies them to intranets. While the book is comprehensive in its coverage of the technology, the coverage is superficial and fails to explain how intranets may change some of the security requirements.

The book discusses user administration, some technical issues, virus detection and prevention, intrusion detection, and some legal issues involved in prosecuting those you discover. The level of detail is suitable for maybe first-line managers, and even some of them may find the detail a little lacking. Even though the discussion of certain issues includes some technical safeguards you can use, details on how to implement the safeguard are completely lacking. Furthermore, the approach of the author is to use off-the-shelf (OTS) software where available. The problem with using OTS software is that it causes additional concerns about its implementation and exactly what safeguards it provides. In addition, you need extra analysis to ensure that the assembly of OTS products satisfies your security requirements. Unfortunately, this book does not discuss how to do that. So the repeated references to commercial products makes the book appear more like an advertisement for those products – an image not helped by the enclosed CD, which includes demonstrations and vendor and product listings.

This commercialism would be more acceptable if the book provided some insight into the new security concerns of intranets. Again it disappoints us, mentioning that intranets raise new concerns but offering nothing to help us understand the problem. Yes, without an external network connection or a very limited connection, the main threat to an intranet is insiders. But what does that mean when we go to secure our network? This book provides few answers to that question.

The superficial coverage of the technical issues makes *Intranet Security* of little value to network or system administrators. It is not worthwhile for readers trying to understand how intranets change the security issues. The comprehensive coverage of general network security issues is worthwhile, but then there are already many such books. The book does provide a good survey of some commercial products, so it would be worthwhile to someone looking for guidance to make a decision about purchasing commercial security products.

Victor R. Volkman (ed.)

C/C++ Treasure Chest

R&D Books. ISBN 0-87930-514-2. Pp. 224. $39.95. Includes CD.

Reviewed by Clif Flynt

<clif@cflynt.com>

One of the features that distinguishes experienced programmers from newbies is the bag of tricks that old-timers carry around in their heads (or on a floppy disk). These tried-and-true code snippets let the old pros go straight to solving a problem, instead of figuring out how to handle the details. We call this code reuse.

The biggest hurdle to code reusability is finding the code you want to use. I suspect that all of us have spent days writing some marvelously clever code, only to have a friend say, "Why didn't you just steal the code from the foo package?"

For the past 15 years, the C/C++ Users Group has collected source code and made it available for other programmers to examine, use, and steal ideas from. The trick has been finding the package that has what you need *now*. *C/C++ Treasure Chest,* compiled by Victor Volkman, makes a big dent in solving this problem. The book contains brief descriptions of literally hundreds of C and C++ programs, libraries, and documentation, indexed by keyword, type of functionality, OS/CPU, and title.

Along with the book is a CD-ROM containing the complete C Users Group code distributions, with HTML indexes to direct users to the package they need. These indexes and HTML pages make the difference between 400 packages of data and 400 packages of information.

This book will pay for itself if you find a single package that saves you an hour. My bet is that you'll find more than one package, be it the genetic-algorithm and neural-network toolkits, the OS-independent screen editors, the JPEG library, SQL package, linked-list library, the MS-DOS TSR tools, or other packages. The range of code available is astounding, and, most important, you can find the code you need.

Jody Leber

NT Backup and Restore

O'Reilly & Associates, 1998. ISBN 1-56592-272-7.
Pp. 320. $29.95

Reviewed by Steve Hanson

<hanson_steve@htc.honeywell.com>

Although this book is aimed at the NT backup administrator, it is primarily a general guide to backup strategy. It gives a good overview of backups – why they are important, how to plan and scale them, and how to select a tool for backup. Most of this material is as applicable to a UNIX network as it is to an NT network. In fact, there is only one chapter devoted specifically to NT backup issues and the reasons why NT is particularly challenging (proper backup of the registry and WINS database, etc.). I was disappointed that these issues were not covered in more depth, since doing NT backup is one of the issues I'm dealing with at work. The book didn't help me much with my issues, such as backing up open NT files.

The book does have many strengths, however. Several tables help the admin to understand how many tape devices and how much network bandwidth will be needed to do backups within a given time window. If your site is anything like mine, getting backups done during an off-hour window is an ever-increasing challenge as disk space grows. The charts help to determine whether that new file server is going to throw you over the edge. Likewise the book gives attention to reviews of a number of commercial NT backup products as well as those that are freely available with the OS. And a feature-comparison chart for commercial backup products is nearly as useful to the UNIX admin as to the NT administrator.

This is the only book on backup that I know of, NT or UNIX. This is surprising – we'd all agree that backup is one of the most important things we do, yet there are very few places to learn how to do it well. Many UNIX and NT admins are finding that those commercial backup products are starting to look good as the size of our backup problem increases. This book is a helpful guide to backup in general and how to make sense of the commercial alternatives available to the NT world. Yet I wished that it went into more detail on some of the pitfalls and how to avoid them. Many topics (such as different backup schedules) are covered, but not really in enough detail to be helpful to the professional who has been doing this for some time. I'd recommend this book to the beginning administrator, and particularly to anyone who is considering the purchase of a commercial backup system for NT. It is even fairly useful to the UNIX administrator, as most of the topics covered apply to both environments. However, the old hand at doing backups isn't likely to gain much from the book, with the possible exception of the capacity charts. Even those may not be very useful to anyone with reasonable math facilities, as they are primarily division tables in disguise.

Thanks to Our 1998 Reviewers!

Here is our honor roll of book reviewers for the past year! Join the crowd: review a book for us!

William S. Annis, Reginald Beardsley, Nick Christenson, Clifton Flynt, Andrew Hume, Rob Jenson, Chris Kottaridis, Daniel Lazenby, George W. Leach, Bruce O'Neel, Terry Rooker, Carolyn J. Sienkiewicz, Kartik Subbarao, and Rick Umali.

Again, thank you!

USENIX news

In Memoriam: John Lions

by Peter H. Salus

John Lions, the principal instigator of UNIX in Australia, founder of the AUUG, and author of the Commentary on V6 (which Ken Thompson called "the best book on how an operating system works"), died on Saturday, 5 December 1998.

John had been quite ill for nearly five years.

John earned degrees from Sydney University and Oxford. He worked for Burroughs in Canada until he went to the University of New South Wales in 1972. He retired in July 1995.

In 1975–76, while teaching a course in operating systems, John wrote a commentary on the source code for his students. The booklets (in bright red and orange covers) were announced in *UNIX NEWS* (the ancestor of *;login:*) and the UKUUG newsletter, only to be suppressed by AT&T lawyers. In 1993, Dennis Ritchie and I attempted to obtain permission to publish these often-xeroxed pamphlets, but only succeeded in 1996, when the rights were sold to SCO; John's masterpiece is finally available. It was reviewed by Jaap Akkerhuis in *Matrix News* 704 (April 1997). The Commentary and Source Code is available as ISBN 1-57398-013-7; the Japanese translation is ISBN 4-7561-1844-5.

Greg Rose (Qualcomm, Australia), who was one of John's students and is now Vice President of USENIX, wrote:

"While at the University of New South Wales, John introduced a course in Operating Systems, and decided to study the UNIX operating system. One of his motivations in doing this was to introduce the students to code which was well written by other people – at the time this was not a common practice, although it is now well accepted – and which implemented a very significant system. In the course of developing notes for this, he wrote an annotation of the source code of UNIX, and produced a pair of books (recently republished and translated) including the source code itself. This was a remarkable achievement, and demonstrated the clarity of thought of which he was capable. The books were not available for general distribution at the time, but were probably the most successful illegally copied books ever; there are numerous reports of '5th generation photocopies'.

"I remember an incident when I was a student of John's, simultaneously helping to run the PDP-11 computer in the then Department of Computer Science, where the computer was often 'locking up' under high load. John took home listings of the current source code, and returned the next morning with details of two race conditions and a potential deadlock in the UNIX kernel which might have explained the problem, and indeed when they were fixed the problem went away.

"John pulled together a group of people interested in UNIX, and when it was later formalized as the Australian UNIX Users' Group became the founding president of the organization.

"Within the University of New South Wales there was a battle over centralization versus distribution of computing resources, which indirectly had a major effect on the autonomy of the Department of Computer Science. John's battle to have UNIX accepted as a vehicle for teaching, and later as the subject of teaching, instrumentally led to the increasing importance and independence of the department. Further, the existence of a centre at UNSW helped the formation of similar groups at Melbourne and Sydney Universities."

John Lions was a fine teacher and a good friend. I am certain that all members of

the Association join me in expressing heartfelt condolences to his widow, Marianne.

Changes

Lately I have been in a contemplative

by Andrew Hume
President, USENIX Board of Directors
<andrew@research.att.com>

mood. There are several good reasons why. I have started mentoring (via email) a female undergraduate at a university in the Midwest. The project I have spent the last 3 1/2 years on is increasingly under its own management and direction. In April, my wife is expecting our first children (twins!). And, worst of all, I am getting really cranky about the software I use every day.

From 1983 to 1996 I worked in computer science research at Bell Labs, beside some of the best in our field (Ritchie, Thompson, Kernighan, McIlroy, Pike, . . .). Even better, I got to use their software! Eighth, ninth and tenth editions of Research UNIX, and the various editions of Plan 9. Best of all, we had Dave Presotto riding herd on the two greatest fiascos of modern computer systems, mailers and networking. During this time I was quite productive in my work, largely due to the quality and smoothness of my computing environment, especially the effective pervasive user interface and window system done by Rob Pike.

Strangely, I was often pitied by outsiders as stagnating in my own insular backwater. I was missing out on the software revolution, the brave new world of the PC with its stunning, fabulous diversity of really keen products to help me do my work and increase my productivity. When I moved from Bell Labs to AT&T Labs, I undertook to experience the mass market full on! I tried to live on the frontier, using Windows 95 and X. Furthermore, my projects involved production systems, so I got to see and use real software (not that fake stuff we had in Research). After three years, I have learned some lessons and thought I'd pass them along. Of course, these are just one person's opinions; I would expect your mileage to vary.

1. As far as I can tell, the only useful outcome of the PC business is high-performance commodity hardware. The software sucks. Even at its worst moments, Plan 9 was more reliable than Windows 95. Of course, it worked okay (meaning it only crashed every week or two) if you didn't do much networking or open many windows or install any new software. But this sounds like giving up to me.
2. But what about the fabulous marketplace of software to do my bidding? Well, I tried that. I wanted some JPEG/MPEG viewers. So I followed the new order and went to <www.tucows.com> and downloaded every freeware/shareware viewer they had. Of the fifteen I tried, five destroyed my Windows software (complete reload required), and eight coredumped or caused the system to crash. But two actually worked! One of these had one of the worst user interfaces I've seen in some time; one was actually usable.
3. So I am giving up on the whole Windows thing. I am switching to Linux or a BSD system; it should be much better, and I can't imagine it being worse.
4. Perhaps more important, I think I am shifting to the open source camp. Perhaps I was unlucky, but in my current project we used five distinct software products on the production machine (other than what we developed). Three were commercial offerings (the system software, a backup suite, and a file transfer suite), and two were supported by folks in AT&T Labs - Research (sorting and searching pro-

grams). After fifteen months, the story was:

- Searching: 0 bugs, high performance
- Sorting: 5 bugs, two months to fix
- Backup: 10+ bugs, several tapes' worth of data lost, mediocre performance, 1 fix delivered during severity-1 mode
- File transfer: 5 serious bugs, 3 fixed by patches within a couple of months, 2 still unresolved after a year
- System software: 10+ serious bugs, all but 1 fixed within 2-5 months

I seem to have come full circle back to where I started in 1975 – running UNIX and software developed by a worldwide collaborative community. And for pretty much the original reasons: it's a very effective way to use the hardware, and the software (and software environment) is simply better. (Of course, I'm still hoping for another release of Plan 9!) Although some part of me is depressed at the apparent lack of progress in 24 years, the rest of me is excited at getting back to an environment where I am much more effective.

Board Meeting Summary

by Ellie Young
Executive Director
<ellie@usenix.org>

Here is a summary of the actions taken at the regular meeting of the USENIX Board of Directors held on November 5-6, 1998, in Berkeley, CA.

Attendance: USENIX Board: Geer, Honeyman, Hume, Parseghian, Pomeranz, Rose, and Zwicky. USENIX Staff/Guests: Young, Berkowitz, Barnett, DesHarnais, Klein, Deno, Long, Appelman, Johnson, Miller, Stoughton, and Suski.

Proposals for Funding

USACO. The proposal to continue funding the USA Computing Olympiad (which selects and trains the US team for the International Olympiad in Informatics) was approved.

Software Patent Institute. A proposal from the Software Patent Institute for a grant of $55,000 to continue to expand and improve the SPI database of software technologies was approved. This database is presently accessible to the public, with the goal of helping patent applicants and the US Patent & Trade Office issue validated patents in the software field.

Incident Cost Analysis & Modeling Project Frequency Study. A request from the University of Michigan for funding support of $64,510 for this project was approved. The project is designed to provide system administrators with additional information regarding which IT related risks are of highest priority to address, about factors relating to occurrence, costs, and how best to manage certain risks, and how to cost-effectively manage incidents. The project also provides a valuable and active learning environment for graduate students wishing to explore the economic as well as technical aspects of IT related events.

USENIX Scholastic Committee. An additional $20,000 was allocated to fund research grants and scholarships for the December 1998 round of applications.

Standards. Stoughton's proposal for a moderate increase in our involvement, covering the new POSIX revision projects, membership in The Open Group, and attending meetings of the TOG Base Working Group was approved. Hall and Stoughton will look into the possibility of funding members of the Linux community to attend meetings.

WEB SITE
http://www.usenix.org

MEMBERSHIP
Telephone: 510 528 8649
Email: <office@usenix.org>

PUBLICATIONS
Jane-Ellen Long
Telephone: 510 528 8649
Email: <jel@usenix.org>

USENIX SUPPORTING MEMBERS
Apunix Computer Services
Cirrus Technologies
Cisco Systems, Inc.
CyberSource Coporation
Deer Run Associates
Hewlett-Packard India Software Operations
Internet Security Systems, Inc.
Microsoft Research
NeoSoft, Inc.
New Riders Press
Nimrod AS
O'Reilly & Associates
Performance Computing Magazine
Questra Consulting
Sendmail, Inc.
TeamQuest Corporation
UUNET Technologies, Inc.
Windows NT Systems Magazine
WITSEC, Inc.

Berkeley Foundation for Opportunity in Information Technology. It was agreed to become a founding donor of this project with a donation of $10,000. BFOIT has been recently formed to assist women and underrepresented minority students enter the IT industry. Assistance will include providing financial aid for students, create relationships with local IT corporations for jobs and internships, and other support as needed for students doing IT coursework in CS and electrical engineering.

Draft Budget 1999

The assumptions behind the first draft budget for 1999 were discussed. It was decided that we should continue to budget conservatively for conference attendance, especially since this is an area that is vulnerable to economic downturns; that we increase the marketing budget for the Security symposium; that we consider increasing fees and dues to handle the increase in the number of events we are sponsoring (going from 8 to 12 in 1999), the rising costs of holding conferences, and the increased costs of serving the membership.

A marketing advisory committee of the Board was formed to look into issues such as brand recognition, how we measure and correlate the results of our marketing efforts, and how we can increase attendance at some of our smaller or newer conferences.

Membership Dues

After discussion about the increased costs of providing services to a greater number of members (while dues have not been raised since 1994), it was agreed to set fees as follows:

	1999	2000
Individual	$80	$95
Supporting	$1000	$1000
	$2500	$2500
Educational	$200	$200
Corporate	$400	$400
Affiliates	$75	$90

Conference Registration Fees

The staff's report revealed that the costs for putting on conferences have skyrocketed in the past couple of years, while USENIX registration fees continue to be among the lowest. It was agreed to raise fees for the next two years as follows:

Tutorials

1999	2000
$395 one day	$445 one day
$690 two days	$740 two days
$985 three days	$1035 three days

Three Days of Tech Sessions

1999	2000
$400	$435

Two Days of Tech Sessions

1999	2000
$360	$410

It was also decided the conference registration fees for students will remain the same ($75), and the non-member conference fees will be the conference member rate plus the cost of an individual membership.

Electronic Voting

It was agreed that Young will look into enabling electronic voting for the next USENIX Board elections, especially exploring what the auditors' recommendations would be regarding tabulating votes electronically.

Privacy Statement

The Board voted to adopt a USENIX policy statement and to publish it in *;login:* (see page 82) and on our Web site.

Staffing

Young reported that there had been a lot of staff turnover recently. She has hired a new publications director, Jane-Ellen Long, to replace Eileen Cohen, who was retiring from full-time work; she has hired a Deputy Executive Director, Gale Berkowitz, to handle operations, SAGE, finances and other projects; two other admin positions for member services and publications/ Web have been filled (by Cami Edwards and Jennifer Radtke), and the conference office has hired a new assistant, Vanessa Fonseca. It is anticipated that additional personnel for marketing and the executive office will be needed in 1999.

Embedded Systems Conferences

It was agreed to co-sponsor this conference with the MIT Media Lab with Michael Hawley and Dan Geer as co-chairs.

In Memoriam

It was agreed to publish tributes to the recent deaths of Jon Postel and John Lions.

BOD Next Meeting

The next meeting will be held February 21 in New Orleans, LA.

20 Years Ago in UNIX

by Peter H. Salus

<peter@pedant.com>

Note another change in title! *;login:* (a.k.a. *UNIX NEWS*) did not reappear until 1980. I wrote last issue that I'd talk about the Association and its activities in 1999. But I'd like to say a bit about UNIX, too.

1978 saw the appearance of Version 7, the first BSD tape (containing Pascal and "the ex editor"), the Ritchie-Johnson port to the Interdata 8, the Wollongong port to the Interdata 7, the port to 32V by Charlie Roberts and a group at Holmdel, the publication of uucp, and the first meeting of the NLUUG – and, of course, the publication of *The C Programming Language* and the "blue" *BSTJ*.

It may be worthwhile to itemize some of the things found for the first time in V7: `intro` (introduction to commands), `adb` (debugger), `at`, `awk`, `calendar`, `cb` (the C beautifier), `cd`, `cu`, `deroff`, `expr`, `f77`

(the Fortran 77 compiler), `join`, `learn`, `lex`, `lint`, `look`, `m4` (a macro processor), `make`, `sed`, `tabs`, `tail`, `test`, `touch`, `true`, `tsort`, `uucp`; `access`, `acct`, `alarm`, `ioctl`, `lseek`, `umask`, and `utime` – in Sections 1 and 2 alone. `popen`, `scanf`, `stdio`, `string` were among the subroutines; and `adventure` and `backgammon` joined `chess` under games. Finally, `man` (which I use nearly every day) was introduced.

On the other side, the beginning of 1979 saw the Internet with 150 hosts and transatlantic and transpacific connections. Original ARPANET hosts had been connected with 55K dedicated lines. The advent of the acoustic modem slowed things down. In 1978-79 I was running a DECwriter II over a 300-baud line to the IBM 360. Twenty years is a long time in technology.

The January 1979 USENIX conference was held in Santa Monica, CA. Mike O'Brien was the program chair. He wrote me: "I remember nothing about that meeting's program except for time I actually spent on stage yapping. There was an early pre-awards ceremony. I don't remember if Steve Holmgren gave his Rubber Chicken award at that time, but I know I gave one.

"Before the conference opened, Dave Yost and Armando Stettner discovered they were kindred spirits. Dave was driving some gigantic American boat of a convertible, which Armando fell in love with, especially when Armando discovered he could cause a hubcap to pop off at will by cornering it sharply. He reciprocated by demonstrating the 'Rockford Maneuver' in the RAND parking lot, using his own rental car.

"James Garner, in 'The Rockford Files,' had a habit of turning his car around by taking off in reverse at top speed, then simultaneously hauling up on the hand brake and slamming the car into Drive. This causes the car to slew around 180 degrees and keep going without appreciable loss of speed.

"What Armando didn't know is that the Santa Monica Police Department is right across the street from RAND. He was beside himself when he noticed this, a few minutes later. I was inspired.

"The next day, I ran off and scored one of the most elaborate T-shirts that the do-your-own-T-shirt shop had ever produced. They photographed it for their archives. It had a giant Cobb cartoon of one of his typical slathering bug-eyed creatures driving a flaming chop job of a race car, surrounded by the words, 'Armando P. Stettner School of Driving.' On the back was the email address of the school, 'rockford!aps'.

"I presented this to Armando at the opening Plenary Session, in front of 1500 or 2000 people. I have never seen Armando in such a state before or since.

"Another thing I remember about Santa Monica are the two innovations I made, one minor, one major. I stole both ideas from DECUS. The minor one was the timer on the speaker's podium, with green, yellow, and red lights. The major one was the USENIX BoF session. In order to level the playing field, I insisted that no reservations for BoF rooms would be accepted before the conference opened. I still believe that that choice was the right one, for I wound up beating off several vendors, who tried to reserve space weeks before the conference. The conference hotel, the Miramar, had limited space, and I wanted to favor the groups that didn't have commercial backing, and hence had no hope of getting meeting facilities outside of a BoF. I think that was the one and only year that policy was enforced, but then it's also been true that BoF space has been a lot more readily available since then.

"Dave Yost's house party, at this conference, may have been the first of the 'wizard' parties. If there was one before this one, I can't remember it offhand, except for Steve Holmgren's kegger at his house at the first east-west meeting in Shampoo-Banana. These have always been more or less elitist affairs. Dave, at that time, lived in a house in the Hollywood Hills. He ran a limousine service to and from a drug store parking lot down the hill on Sunset in order to cut down on parking problems on his own street.

"Dave also provided party favors. He had a box full of uninflated balloons, and a helium tank. Next to the helium tank, he had an oxygen tank, with a big hand-lettered sign that read, '1/3 from this tank (big arrow pointing to the oxygen) and 2/3 from this tank (big arrow pointing to the helium).' He did this because, a week or so before, he and I and some other friends went to Disneyland, where Dave did his level best to get us thrown out of the park. He came across a vendor of helium balloons and loaded up. First we went to the theater-in-the-round exhibit, which at that time was sponsored by AT&T. The young beauty who gave the opening spiel was so perfect-looking that she sort of resembled that shellacked food the Japanese put in restaurant windows. She got well into her spiel when Dave, in a completely heliumed-out voice, yelled from the back: 'That's enough of that stuff, lady! We wanna hear some real Bell System stuff!' The girl folded forward over her podium and sort of gave up for a while.

"Later, Dave was yelling this and that in a helium voice while we were in line at Big Thunder Mountain. Two gorillas in blue blazers were soon seen coming along the line, scowling and giving everyone the eye. We passed inspection.

"Finally, in the parking lot, as we were getting ready to leave, Dave used up the last of the helium, then looked at my friend, who was wearing a white cable knit sweater, and said, 'Gee, I'm sorry, looks like you got spots on your sweater.'

"She looked down and said, puzzled and frowning, 'No I didn't.'

"Dave said 'Oh!' and started taking big deep breaths, 'WHOOP-ah WHOOP-ah WHOOP-ah!'

"So, when Dave gave his UNIX Wizard's party, he made sure to have people mix oxygen with the helium. The other thing I remember at that party was Dennis Ritchie sitting on the floor with his 'happy and bemused' look."

I am gratified to note that the Conference did not seem to "break" the Association's general irreverance.

That 25-27 January was Judy DesHarnais's debut. I guess she wasn't deterred.

The USENIX Privacy Statement

The following statement has been posted on the USENIX Web site. It discloses the USENIX Association's information gathering and dissemination practices.

The USENIX Association maintains member databases that contain mailing, billing, and member profile information, as well as a record of each member's product purchases and registrations for conferences. The information in these databases is used by authorized USENIX staff members to process orders; mail invoices, purchases, renewal notices, and announcements; and respond to member inquiries. Member records are maintained as long as an individual is a USENIX member and for three years following a membership lapse. Purchase and registration databases may be retained for up to three years. USENIX logs all accesses from your IP address to our Web site, and summary information derived from these logs may be displayed on the Web. Personal information is stored in a relational database and standard security methods are used to protect it. We require full confirmation of identity before releasing information back to the user for update over the Web. Commercial information (such as credit card numbers) is obtained using our secure Web server and may be stored on our machines. When the data are stored, we use public key encryption on the sensitive data and require our staff to supply a private key to decrypt it. When passing such information internally over the Internet, we use our secure server.

Electronic Communication

The USENIX Association does not rent or sell email addresses. USENIX may use email addresses to contact members to answer member questions or to acknowledge the receipt of membership applications and other orders, to send membership renewal notices, and to send occasional announcements about USENIX events to those members who have not opted out of receiving such announcements. These announcements are short, straightforward messages that contain pointers to online resources where members can explore the information more fully.

Postal Mail

USENIX uses the postal addresses of its members to mail invoices, products, and announcements from USENIX. The names and mailing addresses of members who have not opted out (see below) may be rented to outside organizations to send mailing packages that have been carefully screened by authorized USENIX staff for their suitability. USENIX member names and mailing addresses are never sold.

Opt-Out

Our Web site gives you the opportunity to opt out either from receiving email communications or from having your name and address made available to anyone other than the USENIX Association. When registering for a conference, you have the option not to be on the Attendee list.

Change/Modify

The USENIX Web site gives users the opportunity to change and modify information previously provided by using the online form found at <www.usenix.org/membership/membership.html>, or by phone: 510 528 8649, fax: 510 548 5738, or email: <office@usenix.org>.

Note: Information is always updated whenever a new conference registration occurs or upon renewal of a membership.

Links

The USENIX Web site contains links to other sites. The USENIX Association is not responsible for the privacy practices or the content of such Web sites.

Contacting USENIX

If you have any questions about this privacy statement, the practices, or your dealings with the USENIX Web site, you can contact the USENIX Association, 2560 Ninth Street, Suite 215, Berkeley, CA 94710.

Thanks to Our 1998 Contributors

Our thanks to the many authors who make *;login:* the most valuable benefit our members receive:

Peter Brundrett, Mark Burgess, Nick Christenson, Phil Cox, Matt Curtin, Justin Dolske, Dan Farmer, Rik Farrow, Dario Forte, Yair Frankel, Daniel E. Geer, Jr., Bob Gray, Neil Gunther, Scott Guthery, Stig HackVän, Joseph N. Hall, Jon Howell, Ping Huang, Chris Lalonde, Elias Levy, David M. Martin, Glen McCluskey, Jeanette McLeod, Jon Meek, Mark K. Mellis, Scott Hazen Mueller, Prithvi Rao, Greg Rose, Peter H. Salus, John Sellens, Mark Sienkiewicz, Daniel E. Singer, Diomidis Spinellis, Nicholas M. Stoughton, Dave Taylor, Warren Toomey, Moti Yung, and Elizabeth Zwicky.

Thanks to our Volunteers

by Ellie Young
Executive Director
<ellie@usenix.org>

USENIX's success would not be possible without the volunteers who lend their expertise and support for our conferences, publications, member services, and philanthropic activities. While there are many who serve on program committees, coordinate the various activities at the conferences, work on committees, and contribute to this magazine, I would like to make special mention of the following individuals who made significant contributions in 1998.

The program chairs for our 1998 conferences:

Avi Rubin, 8th USENIX Security Symposium

Fred Douglis, USENIX Technical Conference

Joe Sventek, 4th Conference on Object-Oriented Technologies & Systems

Don Libes & Michael McLenna, 6th Tcl/Tk Conference

Thorsten von Eicken & Susan Owicki, 2nd USENIX Windows NT Workshop

Remy Evard & Ian Reddy, LISA-NT Conference

Bennet Yee, 3rd USENIX Workshop on Electronic Commerce

Xev Gittler & Rob Kolstad, 12th LISA Conference

The conferences' Invited Talk/Special Track Coordinators:

Clem Cole & Berry Kercheval for the invited talks at the USENIX Conference

Jon "maddog" Hall for organizing the FREENIX track at the USENIX Technical Conference

Doug Schmidt for serving as tutorial program chair for COOTS '98

Dan Geer for organizing the public key infrastructure session at the Electronic Commerce Workshop

Phil Scarr & Pat Wilson for the invited talks at LISA

Adam Moskowitz for organizing the Advanced Topics workshop at LISA

Rob Kolstad & Joel Avery for organizing the Global-LISA workshop at LISA

Dmitry Lenkov for organizing the Advanced Topics Workship at COOTS

Evi Nemeth for organizing student volunteers and MBone services at various conferences

Elizabeth Zwicky for serving as liaison and program committee member for the NLUUG/USENIX SANE conference

Members of the the USENIX Board of Directors who "retired" in Summer '98:

Eric Allman, Margo Seltzer, and Lori Grob

The eight members of the current USENIX Board of Directors who devote so many of their "free" hours providing leadership and governance. A special thanks to Andrew Hume who, as president, has helped in holding things together in what was at times a difficult year.

I would like to thank the SAGE executive committee members whose term expires this February for their contributions:

Tim Gassaway, Hal Miller, Barb Dijker, Kim Trudel, Helen Harrison, Amy Kreiling, and Pat Wilson

The following people provided extra effort with SAGE activities:

Barb Dijker & Mark Lamourine for putting together the "Day in the Life of a System Administrator"

Tim Gassaway, Barb Dijker, and Kim Trudel for serving on the SAGE certification committee

Pat Wilson for chairing the SAGE nominating committee for the Elections and helping with the Web site, especially SAGE Jobs-Offered

And also:

Steve Johnson for serving as the liaison to the Computing Research Association

Eric Allman and Hal Pomeranz for serving as the USENIX board liaisons to the SAGE Executive Committee

Margo Seltzer, Darrell Long, David Kotz, Lori Grob, Pat Parseghian, and Peter Honeyman for serving on the USENIX Scholastic Committee, which oversees the student research grant and scholarship programs

Members of the USENIX Tutorial Review Committee who continue to help the staff in developing our program

USENIX is grateful to all!

USENIX Networking '99
1st Conference on Network Administration
&
1st Workshop on Intrusion Detection and Network Monitoring

Wednesday-Monday, April 7-12, 1999 • Santa Clara, California

Six Days of Sharing Networking Solutions Including Two Days of Tutorials

NETA & ID Tutorial Program Friday-Saturday, April 9-10, 1999

Friday April 9, 1999

Configuring Cisco Routers on an IP Network
William LeFebvre, *Group sys Consulting*

Intrusion Detection and Network Forensics
Marcus J. Ranum, *Network Flight Recorder, Inc.*

Handling Computer and Network Security Incidents
Jim Duncan, *Penn State University*, and Rik Farrow, *Consultant*

How Networks Work: The Limits of Modern Internetworking
Dr. Vincent C. Jones, *PE*

Saturday April 10, 1999

Secure Communications Over Open Networks
Marcus J. Ranum, *Network Flight Recorder, Inc.*

Computer Attacks: Trends and Countermeasures
Tina Darmohray, *SystemExperts, Corp.*;
Phil Cox, *Networking Technology Solutions*

Internet Security for UNIX System Administrators
Ed DeHart, *Pittsburgh OnLine, Inc.*

Topics in Network Administration:

Part 1: IP Addressing for Surviving in the Global Internet.
Howard C. Berkowitz, *Network Architecture Consultant*

Part 2: Faster and Faster – Gigabit Ethernet Networks, File Servers, and Users
Stuart McRobert, *Imperial College, London*

VISIT OUR WEB SITE: *http://www.usenix.org/events/neta99/tut.html*

USENIX Networking '99
1st Conference on Network Administration
&
1st Workshop on Intrusion Detection and Network Monitoring

Wednesday-Monday, April 7-12, 1999 • Santa Clara, California

NETA Technical Program Wednesday–Thursday, April 7-8, 1999

Wednesday, April 7, 1999

Opening Session

Opening Remarks & Awards Paul Ebersman and David Williamson, *Program Co-Chairs*

Keynote Address Norm Schryer, *AT&T Research Labs*

"Home, Road and Work Have Merged Via the Internet"

Folks work all the time, everywhere: at home, on the road and at work. The network is everywhere, both wired and wireless. Provisioning, operating and maintaining such distributed systems opens new universes of services and disasters.

EXAMPLES: Folks listen to music over the data network and don't want it disturbed; if your VPN/IPSEC vendor flunks certificate handling then folks at home and on the road are dead—and so are you.

Standards, interoperability and vendor management are the keys to success and continued survival. Norm's talk will cover the modern world's neat functionality, distributed responsibility and central fragility.

NORM SCHRYER received a Ph.D. in Mathematics from the University of Michigan in 1969 and then joined the Computing Science Research Center of AT&T Bell Laboratories. In 1996, at the AT&T/Lucent/NCR tri-vestiture, he moved to AT&T Labs Research, where he is presently Division Manager of Broadband Services Research: cable modem and optical fiber links to homes and the wireless remote piloting of vehicles/telepresences.

Monitoring and Video Session Chair: Jeff Jensen, *WebTV Networks, Inc.*

Driving via the Rearview Mirror: Managing a Network with Super MRTG
Jeff Allen, *WebTV Networks, Inc.*

Don't Just Talk About the Weather—Manage It! A System for Measuring, Monitoring, and Managing Internet Performance and Connectivity
Cindy Bickerstaff, Ken True, Charles Smothers, Tod Oace, Jeff Sedayao, *Intel Corporation;* and Clinton Wong, *@Home Networks*

Supporting H.323 Video and Voice in an Enterprise Network
Randal Abler and Gail Wells, *Georgia Institute of Technology*

Configuration Management and Security Session Chair: William LeFebvre, *Group sys Consulting*

Network Documentation: A Web-Based Relational Database Approach
Wade Warner and Rajshekhar Sunderraman, *Georgia State University*

Just Type Make! Managing Internet Firewalls, Including Router Access Control Lists, Sendmail Configurations, DNS Databases, and OS Upgrades, Using Make and Other Publicly Available Utilities
Sally Hambridge, Charles Smothers, Tod Oace, and Jeff Sedayao, *Intel Corporation*

Tricks You Can Do If Your Firewall Is a Bridge
Thomas A. Limoncelli, *Lucent Technologies*

New Challenges and Dangers for the DNS Jim Reid, *Origin b.v.*

New and probably unavoidable developments including IPv6, Secure DNS, SRV records and dynamic updates are soon going to have a major impact on DNS administration. Each of these developments poses its own set of problems. Jim will describe these challenges and examine likely solutions and strategies for dealing with them.

USENIX Networking '99
1st Conference on Network Administration
&
1st Workshop on Intrusion Detection and Network Monitoring

Wednesday-Monday, April 7-12, 1999 • Santa Clara, California

Thursday, April 8, 1999

Problems with World-Wide Networking Holly Brackett Pease, *Digital Isle*

Administering a worldwide network presents unique routing and logistics problems. Routing policies set by some country's Internet exchanges, by Tier-1 ISPs, and by in-country providers can sometimes be arbitrary and impossible to predict. In this talk Holly will discuss these policies and offer suggestions for navigating them. She will also cover some less-than-glamorous logistics problems presented by the nature of a global network such as: government carriers, interface standards, and taxes.

The Little NIC That Could Christopher J. Wargaski, *RMS Business Systems*

Usually the creation and operation of a Network Information Center (NIC) is a costly endeavor requiring vast personnel and equipment resources. This can be a difficult task, especially in a large politically charged environment undertaking cost-cutting measures. Using another model, however, a NIC can be created and run in an efficient manner using only a modest amount of new hardware and software resources, and without additional personnel resources.

Splitting IP Networks: The 1999 Update Thomas Limoncelli, *Lucent Technologies*

Thomas Limoncelli will discuss techniques for renumbering and splitting IP networks. These techniques were perfected when splitting AT&T's Bell Labs networks in Holmdel, NJ during the AT&T/Lucent split. Renumbering isn't fun, but it is more common every day. He will focus on their trials and tribulations but emphasize techniques that can be used anywhere. Find out what has been learned since the original presentation at LISA '97.

Network Management on the Cheap Rob Wargaski, *RMS Business Systems*

This presentation discusses the need and utility of a network management system, and recognizes that many organizations are not willing (for a variety of reasons) to invest in one of the "big" systems. Useful tools can be freely obtained, and run on a Linux system. Rob will describe some tactical and strategic tools and show how they can be used to improve the health of a network.

Evolution of VLAN/ELAN Architecture at Vanderbilt University John Brassil, *Vanderbilt University*

John Brassil will examine the design and implementation of VLAN architecture at Vanderbilt University that began as part of the Backbone Reengineering Project (1995-98) and the subsequent changes to that design. Since the backbone is ATM-based and edge networks are Ethernet LANs, the parallel Emulated LAN (ELAN) architecture and its evolution will also be described.

The talk is intended primarily as a case study of VLAN/ELAN implementation in a large university or corporate environment. It will describe the factors which influence design decisions, and the tradeoffs/pitfalls that accompany a particular choice. Design considerations for an MPOA (Multi-Protocol Over ATM) architecture will also be discussed.

Interoperable Virtual Private Networks (VPNs), Directory Services, and Security Eric Greenberg, *Seine Dynamics*

In order to achieve an organization's network application performance and functional objectives, and make for more manageable and effective deployments, an integrated "Network Application Framework" design approach must be taken. Eric Greenberg will address key areas of framework integration: Virtual Private Networks (VPN) including IPSEC, PPTP, and L2TP; Directory Services including LDAP, NDS, and X.500; Single sign-on and network/application security services including Certificates, Kerberos, and SSL/TLS; and integration of disparate networking architectures including TCP/IP, IBM SNA, and NetWare.

Closing Session

Internet Measurements Evi Nemeth, *University of Colorado, Boulder;* k. claffy, *Cooperative Association for Internet Data Analysis at the San Diego Supercomputer Center, UCSD*

MCI and CAIDA (Cooperative Association for Internet Data Analysis) have dedicated passive measurement boxes (OXcMONS's) on the MCI backbone and at key exchange points. This talk will summarize the data seen on these networks including protocol distributions, packet sizes, flow characteristics, network and AS matrices, etc. We will also present data from an active measurement tool, skitter, that has been used to probe the Internet at about 30,000 key server hosts. Attendees will get a feel for the traffic on the Internet and changes in that traffic over the last couple of years.

USENIX Networking '99
1st Conference on Network Administration
&
1st Workshop on Intrusion Detection and Network Monitoring

Wednesday-Monday April 7-12, 1999 • Santa Clara, California

ID Technical Program Sunday–Monday, April 11-12, 1999

Sunday, April 11, 1999

Opening Remarks Marcus Ranum, *Program Chair*

Keynote Address Peter G. Neumann, Principal Scientist, *Computer Science Laboratory, SRI International*

Challenges for Anomaly and Misuse Detection

The field somewhat mistakenly called "intrusion detection" needs to broaden its scope of endeavor in various respects, and overcome some of the characteristic difficulties that have slowed its progress. This talk will address several such approaches:

- Generalizing the domains of detectability to include other aspects such as reliability, survivability, and financial stability
- Providing unprecedented flexibility and interoperability among different analysis systems
- Integrating with other computer-communication technologies such as heterogeneous network management and the Web
- Incorporating robust tamperproofing and modern software engineering into future systems for analysis and response
- Enabling sound dynamic reconfigurability based on analysis results
- Research directions
- The need for robust open-source systems and ongoing testbed environments
- Greater forcing functions needed on developers of operating system components and applications.

Peter G. Neumann has worked on survivability since the mid-1950s, on system security starting with Multics in 1965, on intrusion detection since 1983, and on reliability, safety and risks more recently. He is author of the Addison-Wesley book Computer-Related Risks *and moderates the RISKS Forum newsgroup (comp.risks). He also is a Fellow of ACM, IEEE, and AAAS. He holds a 1961 PhD from Harvard and a 1960 Dr. rerum naturarum Technische Hochschule from Darmstadt. See his Web site at* http://www.csl.sri.com/neumann/ *for Congressional testimonies, RISKS information, and further background.*

Analysis and Large Networks Session Chair: Fred Avolio, *Avolio Consulting*

Analysis Techniques for Detecting Coordinated Attacks and Probes
Tim Aldrich, Stephen Northcutt, Bill Ralph, *Naval Surface Warfare Center Dahlgren Division*

Intrusion Detection and Intrusion Prevention on a Large Network: A Case Study
Tom Dunigan, Greg Hinkel, *Oak Ridge National Laboratory*

An Eye on Network Intruder-Administrator Shootouts
Luc Girardin, *UBS, Ubilab*

Invited Talks Session Chair: Marcus Ranum, *Network Flight Recorder, Inc.*

Why Monitoring Mobile Code Is Harder Than It Sounds Gary McGraw, *Reliable Software Technologies*

Mobile code is code that traverses a network during its lifetime and is able to execute at the destination machine. The idea behind mobile code is actually quite simple—sending around data that can be automatically executed wherever it arrives, anywhere on the network. The problem is this: running someone else's code on your computer is a risky activity. Who is to say what the code might try to do and whether or not its activities will be malicious? This is not a new problem by any stretch of the imagination. In fact, it's really an old problem with a new twist. There are many well-known systems for creating and using mobile code. From a security perspective, Java clearly leads the pack. Monitoring mobile code presents some interesting challenges. First and foremost is the problem of identifying mobile code before it runs. Naive approaches, which include scanning port 80 traffic for the <APPLET> tag, are known not to work. Another problem is determining which resources mobile code should and should not be allowed to access, and making sure the policy is enforced. Complex policy-oriented systems like JDK 1.2 (based on code signing and access control lists) may actually make things harder.

USENIX Networking '99
1st Conference on Network Administration
&
1st Workshop on Intrusion Detection and Network Monitoring

Wednesday-Monday, April 7-12, 1999 • Santa Clara, California

Software and Processes Session Chair: Tina Darmohray, *SystemExperts, Corp.*

On Preventing Intrusions by Process Behavior Monitoring
R. Sekar, *Iowa State University;* Thomas Bowen, Mark Seagal, *Bellcore*

Intrusion Detection Through Dynamic Software Measurement
Sebastian Elbaum, John C. Munson, *University of Idaho*

Learning Program Behavior Profiles for Intrusion Detection
Anup Ghosh, Aaron Schwartzbard, Michael Schatz, *Reliable Software Technologies*

Birds-of-a-Feather Sessions (BoFs)

Monday, April 12, 1999

IDS Systems Session Chair: Charles Antonelli, *University of Michigan*

Automated Intrusion Detection Methods Using NFR
Wenke Lee, Christopher Park, Salvatore J. Stolfo, *Columbia University*

Experience with EMERALD Thus Far
Phillip A. Porras, Peter G. Neumann, Teresa Lunt, *SRI International*

Defending Against the Wily Surfer—Web-Based Attacks and Defenses
Dan Klein, *LoneWolf Systems*

Network Data Processing and Storage Session Chair: Dan Geer, *CERTCO*

Preprocessor Algorithm for Network Management Codebook
Minaxi Gupta, Mani Subramanian, *Georgia Institute of Technology*

The Packet Vault: Secure Storage of Network Data
Charles J. Antonelli, Matthew Undy, Peter Honeyman, *Center for Information Technology Integration, University of Michigan*

Real-Time Intrusion Detection and Suppression in ATM Networks
Ricardo Bettati, Wei Zhao, Dan Teodor, *Texas A&M University*

Invited Talks Session Chair: Norm Laudermilch, *UUNet/Worldcom*

Design and Integration Principles for Large-Scale Infrastructure Protection
Edward Amoroso, *AT&T*

Basic intrusion detection design and integration principles are outlined for practical large-scale infrastructure protection schemes. Issues in the development of middleware for multi-vendor interoperability, algorithms for high-volume alarm processing, and visualization techniques for intrusion display are included.

Experiences Learned from Bro
Vern Paxson, *Network Research Group, Lawrence Berkeley National Labs*

Bro is a system for detecting network intruders in realtime by passively monitoring a network link. Its design emphasizes high-speed (FDDI-rate) monitoring, real-time notification, clear separation between mechanism and policy, and extensibility. To achieve these ends, Bro is divided into an "event engine" that reduces a kernel-filtered network traffic stream into a series of higher-level events, and a "policy script interpreter" that interprets event handlers written in a specialized language used to express a site's security policy. Bro has been in production use since early 1996. We discuss the structure of the system and the lessons learned from our experiences, with an emphasis on some of the key challenges for future intrusion detection systems.

Statistics and Anomalies Session Chair: Marcus Ranum, *Network Flight Recorder, Inc.*

A Statistical Method for Profiling Network Traffic
David Marchette, *Naval Surface Warfare Center, Dahlgren Division*

Transaction-Based Anomaly Detection
Roland Buschkes, Mark Borning, *Aachen University of Technology*

Works-in-Progress Reports (WIPs) Session Chair: Marcus Ranum, *Network Flight Recorder, Inc.*

SAGE-AU'99

The Seventh Annual Conference of The System Administrators Guild of Australia

JULY 5-9, 1999

Sydney, Australia

The SAGE-AU conference is the premier systems administration event in the Asia-Pacific region. The conference offers a premium educational forum for system administrators of all platforms and levels of experience, and an excellent opportunity to meet, network and learn.

Call for Papers and Tutorials

The System Administrators Guild of Australia (SAGE-AU) will be hosting its seventh annual conference in Sydney, July 5-9, 1999.

The annual SAGE-AU Conference, Tutorials and AGM provides a forum for Systems Administrators, Systems Managers, Network Administrators, Developers of Systems Administration Software and Managers of such groups to meet and share their knowledge and experiences, and is the premier event for System Administrators in the Asia-Pacific region.

SAGE-AU'99 is hereby calling for papers and tutorial presentations on any and all topics related to system administration.

Deadlines

Conference paper Summary Abstracts:
Friday 12th February 1999
Submission of tutorial Abstracts:
Friday 12th February 1999
Authors Notified:
Monday 27th February 1999
Conference paper Extended Abstracts:
Friday 2nd April 1999
Works In Progress (WIP) Abstracts:
Friday 21st May 1999
Final papers and tutorial notes:
Friday 4th June 1999

How to Submit

This year's conference will consist of three types of papers, and tutorials:

a) Invited Talks
b) Refereed Papers
c) Works In Progress
d) Tutorials

Timeslots are available for 15, 30, 45 and 60 minute presentations. 5-10 minutes should be reserved for questions from the audience.

15 minute timeslots are Works in Progress.

People presenting a 30+ minute talk will receive free conference registration.

People presenting a 15 minute talk will receive a 50% discount on the conference registration fees.

If you wish to present a paper, send a brief abstract ("Summary Abstract") to the address below by the due date. Please indicate whether you are asking for a 15, 30, 45 or 60 minute timeslot.

Summary Abstracts should be 100 – 200 words in length. Papers should have a technical orientation and should not contain advertising.

Authors of Refereed Papers will be required to submit an Extended Abstract of between 3 and 5 pages This should be sufficient to define the structure of the paper, the major points covered and the areas which require further work. .

Authors giving 30+ minute presentations will be expected to provide a final "camera ready" paper for inclusion in the conference proceedings.

Guidelines to assist authors in planning their presentations will be provided. We encourage those inexperienced in public speaking to trial their papers at the local chapter meetings in order to assist with conference preparation.

Tutorials

Tutorial sessions will be either half day or full day in duration. People wishing to present tutorials should submit an abstract of the material they wish to present, and who their intended audience is. Tutorials should be run in lecture format. Suggested topics include: Computer and Network Security/ Network Authentication PC/Apple/Unix/ Mainframe Interoperability NFS/ Automount/AMD Configuration and Operation Perl/Java/Tcl/Python Sendmail/Qmail/ smapd/Anti-SPAM WWW Cache/Router Config/Firewall Setup/Squid NT/Win95 Administration

Tutorial presenters will be paid $500 for a half day tutorial and $1000 for a full day tutorial. SAGE-AU will reimburse tutorial presenters all reasonable costs of handout materials or will print them on your behalf.

As with papers, tutorials should have a technical orientation and should not contain advertising.

Conference Details

Monday 5th July to Wednesday 7th July, 1999: tutorials on tools and techniques to aid system administration. The AGM will be held at the end of the fourth day (Thursday 8th). All other times will be allocated to presentations or discussions. A conference dinner will be held on Thursday evening.

Exhibition/Trade Show

Wednesday and Thursday July 7-8 SAGE-AU'99 will host a small, technically orientated trade show focusing on system administration tools and information.

Please contact the organisers for details.

Registration

Conference registration includes the Conference Dinner and Conference and Tutorial registration includes Lunch and Refreshments.

Non-members who register for SAGE-AU'99 at the non-member rate and successfully apply for membership of SAGE-AU will have their first year's membership fee waived.

Registration forms for tutorials will be available approximately six weeks before the conference date.

Early Registration is considered when registration form and payment has reached SAGE-AU by COB on 21st May 1999.

Travel

To encourage interstate attendees, SAGE-AU offers members a travel discount off registration for interstate travellers (Tas/WA/SA/NT).

Addresses

Please submit all tutorial and paper abstracts to: *papers@sage-au.org.au*

Media requests for information and general conference enquiries should be directed to: Geoff Halprin SAGE-AU'99 Conference Chair Email: *conference@sage-au.org.au*

Requests for general information about SAGE-AU and membership applications should be addressed to:
WWW: *http://www.sage-au.org.au/*
Email: secretary@sage-au.org.au
Fax: 0500 544 488 (Attn: David Conran)
Snail: Secretary SAGE-AU GPO Box 2974 Sydney NSW 2001 Australia

The web page for the conference is *http://www.sage-au.org.au/conf.html*

Call for Papers USENIX

5th Conference on Object-Oriented Technologies and Systems (COOTS '99)

Sponsored by The USENIX Association **Conference Web site: *http://www.usenix.org/events/coots99***

May 3-7, 1999
San Diego, California, USA

Important Due Dates

Paper submissions: *Nov. 6, 1998*
Tutorial submissions: *Nov. 6, 1998*
Notification to authors: *Dec. 16, 1998*
Camera-ready papers: *March 23, 1999*

Conference Organizers

Program Chair
Murthy Devarakonda, *IBM T.J. Watson Research Center*
Program Committee
Ken Arnold, *Sun Microsystems, Inc.*
Rachid Guerraoui, *Swiss Federal Institute of Technology*
Jennifer Hamilton, *Microsoft Corporation*
Doug Lea, *SUNY Oswego*
Gary Leavens, *Iowa State University*
Scott Meyers, *Software Development Consultant*
Ira Pohl, *UC Santa Cruz*
Rajendra Raj, *Morgan Stanley & Company*
Doug Schmidt, *Washington University*
Joe Sventek, *Hewlett-Packard Labs*
Steve Vinoski, *IONA Technologies, Inc.*
Werner Vogels, *Cornell University*
Jim Waldo, *Sun Microsystems*
Yi-Min Wang, *Microsoft Research*
Jack C. Wileden, *University of Massachussetts, Amherst*
Shalini Yajnik, *Bell Laboratories, Lucent Technologies*
Tutorial Program Chair
Douglas C. Schmidt, *Washington University*

Overview

As the last COOTS before the year 2000, COOTS '99 will focus on "The Object Lessons," our cumulative experiences in building and programming object-oriented systems. We invite you to submit high quality, previously unpublished, original papers on this theme as well as on all topics relating to object-oriented systems.

In addition to experience-centered papers, COOTS '99 accepts papers on a wide range of topics, including but not limited to:

Distributed Objects
Object-oriented systems performance
Security for Distributed Objects
Object services
Mobile objects
Object-oriented design technigues
Component based operating systems
Standard Template Library
Advanced C++ topics/examples
Java and Web programming languages
Container technologies (e.g. Java Beans)
Design patterns
Visual J++ and other development tools
Fault tolerance
New OO programming languages
Object-Oriented database systems
Building distributed applications
Persistent Object Issues
Groupware
Patterns
Major fractures of C++
C++
SmallTalk systems
Commercial toolkits/OBDMS
Platform-independent features of C++

Keynote and Invited Speakers

In the long-standing tradition of USENIX conferences, COOTS '99 will feature two prominent speakers, who combine extraordinary insights, original thinking, creativity, and years of experience to make a difference in the way we build and program computer systems. The first day of the technical sessions will feature a keynote address

by James Gosling of Sun Microsystems, and the second day of the technical sessions will feature an invited talk by Professor Barbara Liskov of MIT.

Tutorials

The COOTS conference will begin with two days of tutorials. We expect tutorial topics to include: Distributed object systems (CORBA, DCOM, RMI, etc.), Java and WWW programming languages, framework design, and object-oriented programming languages.

If you are interested in proposing a tutorial, contact the USENIX tutorial coordinator, Dan Klein, by phone at +1.412.422.0285 or by email to **dvk@usenix.org**

Technical Sessions

Two days of technical sessions will follow the tutorials. COOTS emphasizes research and advanced engineering aspects of object technology, focusing on experimental systems research. Conference Proceedings will be published by USENIX and provided free to technical session attendees. An award will be given for the best student paper at the conference.

Advanced Topics Workshop

As usual, the conference will conclude with an Advanced Topics Workshop, where a smaller audience can exchange in-depth technical information on a few position papers. The topic for the ATW will be announced several months before the conference.

What to Submit

Full papers should be 10 to 15 pages (around 5,000-6,000 words). All submissions will be judged on originality, relevance, and correctness.

Each submission must include a cover letter stating the paper title, the contact author, email and regular addresses, and a phone number.

The COOTS conference, like most conferences and journals, requires that papers not be submitted simultaneously to another conference or publication and that submitted papers not be previously or subsequently published elsewhere. Additional information and detailed guidelines for submission and examples of extended abstracts can be obtained by sending email to **coots99authors@usenix.org** or by telephoning USENIX at 510.528.8649.

Where to Submit

Please send one copy of a full paper to the program committee via email (Postscript, PDF, or ASCII) to: **coots99papers@usenix.org**. All submissions will be acknowledged.

Registration Materials

Materials containing all details of the technical and tutorial programs, registration fees and forms, and hotel information will be available in February 1999. Please go to the conference Web site:
http://www.usenix.org/events/coots99

If you would like to receive the program materials in print, contact:

USENIX Conference Office
22672 Lambert Street, Suite 613
Lake Forest, CA USA 92630
Phone: 714.588.8649
Fax: 714.588.9706
Email: conference@usenix.org

Announcement and Call for Participation USENIX

Tcl/2K: The 7th USENIX Tcl/Tk Conference

Sponsored by USENIX, The Advanced Computing Systems Association

http://www.usenix.org/events/tcl2k

February 14-18, 2000 **Marriott Hotel, Austin, Texas**

Important Dates:

Paper, demonstration, and panel proposals due: *September 1, 1999*
Acceptance notification: *October 1, 1999*
Poster submissions: *December 8, 1999*
Camera-ready copy: *December 20, 1999*

Conference Organizers:

Conference Co-Chairs:
De Clarke, *UCO Lick Observatory*
Tom Poindexter, *Talus Technologies Inc.*
Program Committee:
Steve Ball, *Zveno Pty Ltd*
Dave Beazley, *University of Utah*
Dave Griffin, *Compaq Computer Corporation*
Mark Harrison, *AsiaInfo Computer Networks (Beijing), Ltd.*
Melissa Hirschl, *Scriptics Corporation*
Jeffrey Hobbs, *Siemens AG*
Jim Ingham, *Cygnus Solutions*
Michael Johnson, *Pixar Animation Studios*
Cameron Laird, *Network Engineered Solutions*
Don Libes, *NIST*
Michael McLennan, *Lucent Technologies*
Matt Newman, *Sensus Consulting Inc.*
John Reekie, *UC Berkeley EECS*
Mark Roseman, *Teamwave Software Ltd*

The 7th USENIX Tcl/Tk Conference is a forum to:

- bring together Tcl/Tk researchers and practitioners
- publish and present current work involving Tcl/Tk
- learn about the latest developments in Tcl/Tk
- plan for future Tcl/Tk related developments

The conference program will include formal paper and panel presentations, poster and demonstration sessions, works in progress (WIP) sessions, Birds of a Feather (BOF) sessions, and tutorials.

Overview and Forms of Participation

All forms of participation provide an opportunity to report on original Tcl/Tk research. Formal papers should address topics of interest to experienced Tcl/Tk programmers; posters and informal demos may be geared to any level of user from beginner to expert.

Topics include, but are not limited to:

- Tcl/Tk extensions
- Novel Tcl/Tk-based applications
- Experience reports: designing, building, and supporting Tcl/Tk applications
- Comparative evaluations: Tcl/Tk vs. other languages or toolkits for building applications
- Programming paradigms: Those you've used with Tcl/Tk, and proposals for new directions.
- Mission-critical applications: deployment and management of large/critical Tcl/Tk applications

Best Paper Awards

Awards will be given for the best paper and best student paper at the conference.

Papers

Papers should be concise. Omit extraneous or redundant text. Length is not a direct factor in judging paper quality; however, historically, most papers are 12 pages or less. Consider trimming longer papers, possibly by splitting into separate and more focused papers. Authors of accepted papers will have twenty minutes to present the paper at the conference. Submissions for review must be full papers, written in English. Authors are encouraged to include black-and-white figures in their papers.

The program committee will review and evaluate papers according to the following criteria:

- Quantity and quality of novel content
- Relevance and interest to the Tcl/Tk community
- Quality of presentation of content in the paper
- Suitability of content for presentation at the conference

Papers should be 8–12 single-spaced pages and should present a cohesive piece of work. Papers so short as to be considered extended abstracts will not receive full consideration. Papers may report on commercial or non-commercial systems, but those with blatant marketing content will not be accepted.

Application and experience papers need to strike a balance between background on the application domain and the relevance of Tcl/Tk to the application. Application and experience papers should clearly explain how the application or experience illustrates a novel use of Tcl/Tk, and what lessons the Tcl/Tk community can derive from the application or experience to apply to their own development efforts.

This conference requires that papers not be submitted simultaneously to other conferences or publications, and that submitted papers not be previously published or accepted papers subsequently published elsewhere within a period of one year after acceptance. Papers accompanied by non-disclosure agreement forms will be returned to the author(s) unread. All submissions are held in the highest confidentiality prior to publication in the Proceedings, both as a matter

of policy and in accord with the U.S. Copyright Act of 1976.

Posters

Poster submissions provide an opportunity to present interesting or preliminary results. They are the ideal category for material that is better suited for discussion in small groups.

Posters will be displayed during one day of the conference. A poster session will provide an opportunity for attendees to interact with poster authors individually and in small groups as opposed to large groups. Display space will be approximately 3 feet wide by 4 feet high. Poster authors should submit a draft of their poster's contents along with a one-page abstract. Abstracts of accepted posters will be published in the conference proceedings.

Demonstrations

There will be a demonstration reception one evening. Demonstrations will be held in parallel, allowing attendees to more closely interact with the demonstrators. Space will be available for demonstrations in the following categories:

Reviewed demonstrations

- will be given a demonstration station for the entire session and
- will have an abstract published in the conference proceedings.

Submissions should include both a one-page abstract and six copies of a videotape (VHS) showing the demonstration. Some demonstrations may also be scheduled for a conference session.

Informal demonstrations

- will be assigned a specific time during the demonstration session.

Authors of accepted papers as well as those with demonstration-ready Works-in-Progress are encouraged to sign up for informal demonstration time slots. More information on the facilities available for informal demonstrations will be provided in the registration materials and on the conference Web site.

Demonstrations of commercial products of interest to the Tcl/Tk community are encouraged. The abstract for the proceedings, however, should avoid commercial content (i.e., it should not include pricing and sales information or marketing content).

Panel Proposals

The program committee is organizing panel discussions of up to 90 minutes. Proposals should include a list of confirmed panelists, a title and format, and a panel description with position statements from each panelist. Panels should have no more than four speakers, including the panel moderator, and should allow time for substantial interaction with attendees. Panels are not presentations of related research papers. Papers should be submitted individually, and the program committee will group them into sessions of related material.

Works-in-Progress Presentations and Birds-of-a-Feather Sessions

Works-in-Progress (WIP) presentations and Birds-of-a-Feather sessions (BOFs) are not reviewed. Slots for both are available on a first-come, first-served basis starting in January 2000. Specific instructions for reserving WIP and BOF time slots will be provided in the registration information in November 1999. Some WIP and BOF time slots will be held open for on-site reservation, so we encourage all attendees with interesting work in progress to consider presenting that work at the conference.

Tutorials

On Monday and Tuesday, February 14–15, USENIX's well-respected tutorial program will offer intensive, immediately useful, half- and full-day sessions. Skilled instructors who are hands-on experts in their topic areas present both introductory and advanced tutorials.

How to Submit a Paper, Demonstration, or Panel Proposal

We are accepting most conference submissions electronically, via email. Paper, poster, panel and demonstration proposal submissions should be sent in two forms:

- Postscript or PDF files formatted for an 8.5 x 11 inch page with 1 inch margins. (Be sure that it will print on a variety of printers.)
- Plain text or HTML (standard markup only, no browser-specific tags).

Send submissions to:

tcl2kpapers@usenix.org

If accepted, both electronic and camera-ready hardcopy of the final version (full paper, poster abstract, or panel summary and position statements) will be required. Detailed submission instructions will be available by February 1999 at the conference web site at *http://www.usenix.org/events/tcl2k*

Formal demonstration proposals should also include six copies of a VHS videotape showing the demonstration. The videotapes are for review purposes only, and cannot be returned. If accepted, both camera-ready and electronic versions of the abstract will be required.

Postal Address:

Tcl/2K (Tcl/Tk 2000) Conference
USENIX Association
2560 Ninth Street, Suite 215
Berkeley, CA 94710
Phone: 510.528.8649

More information on the conference will be available at the conference web site: *http://www.usenix.org/events/tcl2k*

Registration Materials

Materials containing all details of the technical and tutorial programs, registration fees and forms, and hotel information will be available in November 1999. If you wish to receive the registration materials, please contact:

USENIX Conference Office
22672 Lambert Street, Suite 613
Lake Forest, CA 92630
949.588.8649
Fax: 949.588.9706
Email: *conference@usenix.org*

ASA/MA '99

Agent Systems and Applications/Mobile Agents

IEEE COMPUTER SOCIETY IEEE

Joint Symposium ASA/MA'99 First International Symposium on Agent Systems and Applications (ASA'99) and Third International Symposium on Mobile Agents (MA'99)
Featuring the Third Dartmouth Workshop on Transportable Agents (DWTA'99)
October 3-6, 1999 • Marriottís Rancho Las Palmas • Palm Springs, California
http://www.generalmagic.com/asa
Sponsored by IEEE Task Force on Internetworking - cosponsored by the USENIX Association
In Cooperation with the International Federation for Information Processing (IFIP/TC6)
With Generous Contributions from: Dartmouth College and General Magic, Inc.

In the age of information overflow, agents have become an important paradigm. Agents can act on behalf of users to collect, filter and process information. They can act autonomously and react to changing environments. Agents appear in diverse forms, such as intelligent agents, multi-agents, mobile agents, Internet agents, and manufacturing agents. They are deployed in different settings, such as industrial control, Internet searching, personal assistance, network management, games, software distribution, and many others. The need for agents seems obvious, but a full understanding of their capabilities and limits is an open research topic in numerous communities. This symposium will showcase advanced R&D work in agent technologies. The symposium will focus on experimental research and experiences gained while developing and using agents that meet real user needs. Each submission will be encouraged to make the source code available, and if possible, to demonstrate the technology. We expect to attract submissions from different communities. ASA'99 and MA '99 are being held jointly with one Program Committee and a single technical track. DWTA '99 is a separate workshop held the day after the main symposium.

SYMPOSIUM TOPICS

- agent systems
- agent applications
- tools for agent development
- security for agents
- communication, collaboration and coordination
- mobility of agents
- agent languages
- agents in electronic markets and commerce
- agent societies and ensembles
- constraining agent behavior/resource management
- agent standards
- agent design patterns

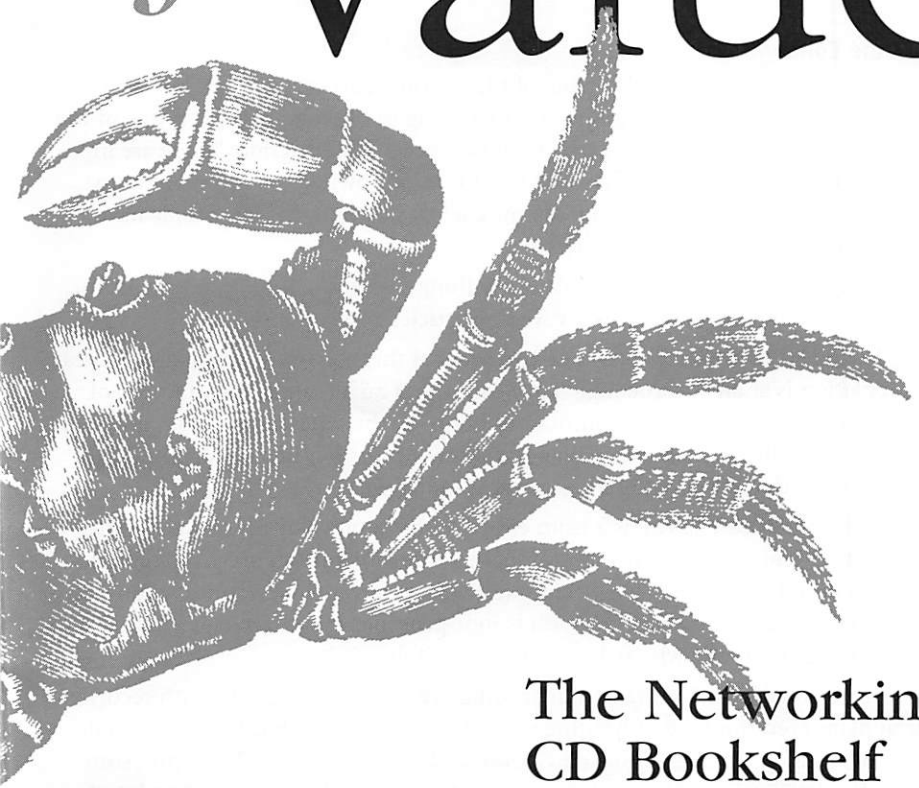
IMPORTANT DATES

- Paper submissions due -- May 3, 1999
- Tutorials and Exhibition proposals -- June 15, 1999
- Notifications to authors -- June 15, 1999
- Camera-ready final papers -- August 15, 1999
- Symposium -- October 3-6, 1999

SUBMISSIONS

Send your complete manuscript (not to exceed 15 single-spaced pages of text using 12-pt type on 8 1/2x11 or A4 pages) to the Program Chair. Electronic submissions are encouraged and should be sent to asa@hpl.hp.com, or made available on the Web. Submissions must be in the form of a readable postscript file. Please, attach also title, author name(s), postal address, e-mail address, and telephone and fax numbers. All papers will be reviewed for quality and relevance to the symposium. Very similar papers must not have been published or concurrently submitted for publication elsewhere. For tutorial proposals, please contact Jean-Pierre.Briot@lip6.fr. For exhibition proposals, please contact cheyer@ai.sri.com. The organizers of ASA/MA '99 and a related conference, the 2nd USENIX Symposium on Internet Technologies and Systems (USITS) http://www.usenix.org/events/usits99, recognize that some papers are appropriate for either conference. By special arrangement, a single paper can be submitted to both conferences, and then withdrawn from ASA/MA '99 if accepted at USITS. Additional information and detailed guidelines for submission can be obtained from http://www.generalmagic.com/asa.

A Different Kind of Value



The Networking CD Bookshelf

The Networking CD Bookshelf is a powerhouse of critical information for network administrators. Six classic O'Reilly books are on the CD, including complete, unabridged versions of *TCP/IP Network Administration, 2nd Ed.; sendmail, 2nd Ed.; sendmail Desktop Reference; DNS and BIND, 3rd Ed.; Practical UNIX & Internet Security, 2nd Ed.;* and *Building Internet Firewalls.* As a bonus, the new hardcopy version of *DNS and BIND, 3rd Ed.* is also included.

Never has it been easier to learn, or look up, what you need to know online. Formatted in HTML, *The Networking CD Bookshelf* can be accessed with any Web browser. The books are fully searchable and cross-referenced. In addition to individual indexes for each book, a master index for the entire library is provided.

All this for an incredible price of $79.95. It's sure to keep anyone from being crabby.

O'REILLY®

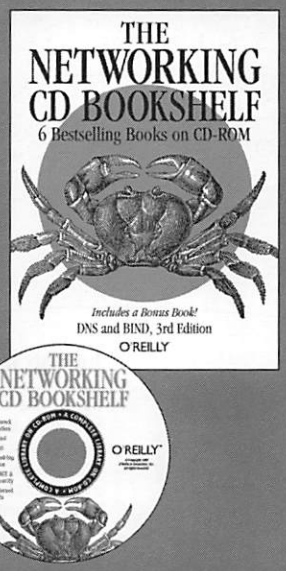
a different kind of animal

101 MORRIS STREET, SEBASTOPOL, CA 95472
WWW.OREILLY.COM

ORDERS/INQUIRIES: **800-998-9938**
WEEKDAYS 6AM-5PM PST
707-829-0515 • FAX: 707-829-0104

EMAIL FOR OUR CATALOG:
CATALOG@OREILLY.COM
INCLUDE YOUR NAME AND MAILING ADDRESS

O'REILLY BOOKS ARE ALSO AVAILABLE AT YOUR BOOKSTORE



The Networking CD Bookshelf includes on CD complete, unabridged versions of:

- ***TCP/IP Network Administration, 2nd Ed.***
- ***sendmail, 2nd Ed.***
- ***sendmail Desktop Reference***
- ***DNS and BIND, 3rd Ed.***
- ***Practical UNIX & Internet Security, 2nd Ed.***
- ***Building Internet Firewalls***

And as a bonus, the hardcopy version of:

- ***DNS and BIND, 3rd Ed.***

PLEASE MENTION CODE
A9LOG
WHEN ORDERING

motd

by Rob Kolstad

Rob Kolstad is president of BSDI and a long-time USENIX member, having served as chair of several conferences and workshops, director on the Board, and editor of *;login:*. He is also head coach of the USA programming team.

<kolstad@usenix.org>

State of the Editor

I watched our (current) President's State of the Union Address last night; he had a whole grab bag of ideas and programs. It's interesting to observe that this is one of the few times that we get to observe a certain kind of leadership in action: "Here are my ideas; let's implement them." While we all see this in our own lives at work, at home, and maybe at other activities like churches or clubs, I just love it when people have great ideas and then go implement them.

I thought maybe I could write a bit about how things are going for *;login:* and for me. With any luck, I'll inspire new ideas for writing articles and columns.

Looking backward at the last year, it's easy to see that this was our Magazine's best year ever. Circulation is at an all-time high; page count is at an all-time high; number of issues, likewise. Editorial quality continues to improve; the typesetting "look" and overall appearance are better than ever. The number of authors has increased along with the variety of articles. It was truly a banner year for *;login:*!

Of course, this is because *;login:* is a team effort. Many people pay lip service to this concept, but it's really true for this job. We have editors for various areas of content, "goaders" who solicit articles, typesetters, guest editors, proofreaders, production people – and, of course, authors. Each person is indispensable. It's a marvelous machine to watch. Thanks to everyone who helped with ;login: this year!

On a more personal note, I've had an interesting year. I'm working now with security people and am even thinking of heading up an initiative to set a bar for a reasonable standard of security. I moved into a new house and was able to "touch" all my "stuff" as it went by after over ten years in the previous house. My oh my but that's a lot of stuff! It's nice out here in the country, though: very quiet and very dark at night. Come visit if you're in the neighborhood (Colorado Springs).

What of the future? What of initiatives and programs to improve the newsletter? USENIX Executive Director Ellie Young has been a tremendous source of new ideas and inspiration for the magazine. She was a driving force in getting the comprehensive conference summaries going. I hope to be able to find information even half as useful.

The page count of *;login:* is about as large as it can get. If larger, we must change binding methods (which should properly be a one-way street) and add a few days to the editorial process. Currently, we turn an entire issue of *;login:* in just under two months - an extremely timely turn. If we continue to get ever more material, we will increase the number of issues per year over the six regular and several special issues already scheduled.

The "look" of *;login:* should continue to improve as we get better at finding relevant photographs and other visual devices. I found a source for crossword puzzles; let me know if you think this issue's puzzle is too hard or too easy.

Our editorial board meeting identified a few new topics we'd like to cover. I'd like to hear from you about this, too. Please write me at <kolstad@usenix.org> if you have ideas about articles or columns you'd like to see.

I'm looking forward to an even better year in 1999. Please do let me know if you'd like to contribute; there's always room!

CONNECT WITH USENIX

MEMBERSHIP AND PUBLICATIONS

USENIX Association
2560 Ninth Street, Suite 215
Berkeley, CA 94710
Phone: 510 528 8649
FAX: 510 548 5738
Email: <office@usenix.org>

WEB SITE

http://www.usenix.org

EMAIL

login@usenix.org

COMMENTS? SUGGESTIONS?

send email to jel@usenix.org

CONTRIBUTIONS SOLICITED

You are encouraged to contribute articles, book reviews, photographs, cartoons, and announcements to *;login:*. Send them via email to <login@usenix.org> or through the postal system to the Association office.

Send SAGE material to <tmd@usenix.org>. The Association reserves the right to edit submitted material.

The closing dates for submissions to the next two issues of *;login:* are April 6, 1999, and June 1, 1999.

USENIX The Advanced Computing Systems Association

;login:

PERIODICALS POSTAGE
PAID
AT BERKELEY, CALIFORNIA
AND ADDITIONAL OFFICES

USENIX Association
2560 Ninth Street, Suite 215
Berkeley, CA 94710

POSTMASTER
Send Address Changes to *;login:*
2560 Ninth Street, Suite 215
Berkeley, CA 94710